⑲日本国特許庁（JP）　⑪特許出願公開

⑫公開特許公報（A）　昭63－127334

| ⑤Int.Cl.⁴ | 識別記号 | 庁内整理番号 | ㊸公開　昭和63年(1988)5月31日 |
|---|---|---|---|
| G 06 F 9/06 | 330 | A－7361－5B | |
| 12/14 | 320 | F－7737－5B | |
| 15/16 | 320 | P－2116－5B | |
| | 420 | S－2116－5B | 審査請求　未請求　発明の数 2　（全36頁） |

㊹発明の名称　ソフトウエア保護機構から安全に実行権を取出し、及び条件付けする方法

㉑特　願　昭62－250063
㉒出　願　昭62(1987)10月5日
優先権主張　㉜1686年11月5日㉝米国(US)㉛927299

㉒発明者　リーム・デヴイド・カマーフオード　アメリカ合衆国ニューヨーク州カーメル、ヴアリイー・ロード、ロード1、ボツクス191番地
㉒発明者　ステイブ・リチヤード・ホワイト　アメリカ合衆国ニューヨーク州ニューヨーク、アパートメント33、パーク・アヴエニユー7番地
㉑出願人　インターナシヨナル・ビジネス・マシーンズ・コーポレーション　アメリカ合衆国10504、ニユーヨーク州　アーモンク（番地なし）
㉔代理人　弁理士　山本　仁朗　外1名

明　細　書

1.発明の名称　ソフトウエア保護機構から安全に実行権を取出し、及び条件付けする方法

2.特許請求の範囲

(1)　上位計算機に関連する論理的に安全な、少なくとも第1のキー及び特定のアプリケーションを実行する権利を表わす第2のソフトウエア・キーを記憶するコプロセツサを有し、上記上位計算機が上記第2のソフトウエア・キーによつて暗号化キーの下に暗号化された上記特定のアプリケーションにアクセスできるソフトウエア保護機構の上記コプロセツサから実行権を取出すための方法であつて、

(a)　上記コプロセツサに、少なくとも書込み可能な媒体及び物理的及び論理的に安全な、平文のトークン・データを記憶する媒体を与え、

(b)　上記コプロセツサに上記第1のキーで暗号化された上記平文のトークン・データより成るデータ・ブロックを与え、

(c)　上記データ・ブロックを上記コプロセツサ中で解読して、上記平文トークン・データを発生し、

(d)　上記平文のトークン・データを上記第2のソフトウエア・キーで暗号化して、対応するデータ・ブロックを発生し、

(e)　上記ソフトウエア・キーを上記第1のキーによつて暗号化して暗号化したソフトウエア・キーを発生し、

(f)　上記対応するデータ・ブロック、暗号化したアプリケーション及び上記暗号化したソフトウエア・キーを上記転送セットに書込んで、上記ソフトウエア・キーを上記コプロセツサから抹消し、

以て上記ソフトウエア・キーを上記コプロセツサから除去して上記転送セットに書込む段階を有する、

ソフトウエア保護機構から実行権を安全に取出す方法。

(2)　ソフトウエア・キーによつて暗号化された部

分を少なくとも含む特定の保護されたアプリケーションを上位プロセッサ並びに物理的及び論理的に安全なコプロセッサを含む複合計算機上で、実行するための、上記コプロセッサの永久メモリ中に記憶されている上記ソフトウエア・キーによつて表わされた実行権に条件を付する方法であつて、

(a) 上記コプロセッサに少なくとも1つの条件のステートメントを与え、

(b) 上記永久メモリ中に、上記ソフトウエア・キーの記憶とともに、上記条件に関連するデータを記憶し、

(c) 上記ソフトウエア・キーもしくは上記保護アプリケーションの使用を許可する前に上記ステートメント及び上記データにアクセスすることを上記コプロセッサに要求し、

(d) さらに上記コプロセッサに上記ステートメント及び上記データを比較し、上記条件が満足されているかどうかを要求し、上記条件を満足している時にのみ使用を許可する段階を有する、

保護されたアプリケーションの実行権に条件を付ける方法。

3.発明の詳細な説明

以下の順序で本発明を説明する。

A　産業上の利用分野

B　従来技術

C　発明が解決しようとする問題点

D　問題点を解決するための手段

E　実施例

E1　ソフトウエア資産保護機構（第2図）

E2　実行権の制約（第1図、第3図、第4図）

E3　実行権の転送（第5図、第6図、第7図、第19図）

E4　実行権のバックアップ（第8図、第9図、第10図、第11図、第12図、第13図、第14図、第15図、第16図、第17図）

E5　販売者のキーの暗号化（EVK）（第18図）

E6　ソフトウエアの返品

F　発明の効果

## A　産業上の利用分野

本発明はデータ処理、具体的にはソフトウエア保護機構に関する。この機構は磁気媒体もしくは他の媒体によつて頒布されるソフトウエアを許可された物理的に安全なコプロセッサに関連する任意の計算機上で使用するように制限する。この機構はユーザのバックアップ・コピーの作成を妨げないが、その保護はこのようなバックアップ・コピーによつて危くされることはない。本発明は特にコピー防止機構の顕著な特徴である実行権の処遇に関する。

## B　従来技術

関連文献としては、本出願人に係る次の米国特許出願明細書があげられる。

米国特許出願第927309号（1986年11月5日出願）

米国特許出願第927306号（1986年11月5日出願）

米国特許出願第927629号（1986年11月5日出願）

米国特許出願第927298号（1986年11月5日出願）

米国特許出願第927286号（1986年11月5日出願）

米国特許出願第927297号（1986年11月5日出願）

基本的なコピー防止機構は上述の米国特許出願第927629号に開示されている。この機構は保護されるべきソフトウエアをこのソフトウエアの実行権から分離している。安全を与え、上記機構を具体化するために、保護されたアプリケーションを実行する各計算機（以下上位計算機と呼ぶ）は論理的及び物理的に安全なコプロセッサと関連付けられる。コプロセッサ中に設置される時は、特定の保護されたアプリケーションの実行権はアプリケーション・キー（AK）と呼ばれるソフトウエア解読キーの形で存在する。ソフトウエア解

読キーＡＫがコプロセッサの永久メモリに保持されている限り、これに対応する保護されたソフトウエアは上位計算機及びコプロセッサを含む複合システム上で実行できる。このソフトウエアのコピー防止機構は現存の及び計画中のソフトウエア頒布技術にほとんど抵触せず、ユーザに無制限のバックアップ（補助）コピーの作成を可能にし、ユーザとソフトウエアの販売者間に２方向通信を必要としないという利点がある。この利点はハードウエア・カートリッジ（即ちトークン）の形で与えられる、実行権を受取るための許可をコプロセッサに頒布することによつて支援される。さらにユーザは未使用のトークンによつて表わされる実行権をコプロセッサに転送するため、保護されたアプリケーションを実行する始めにトークンを使用するだけでよい。その後、トークンは廃棄され、その後はこの実行権（トークン）の保持もしくは使用は全く必要がない。

上記米国特許出願第９２７６２９号に開示されている発明は（ユーザが最初どのようにして実行した時の頒布媒体と同じである）の使用によつて間接的か、コプロセッサからコプロセッサへの通信リンクを介して直接的に行うことができる。尚、このとき、転送トランザクションが傍受されるかも知れないという意味で通信が安全でない時でも安全が保持される。

本発明は又実行権に条件を付けるための技術を与える。たとえば実行権は期間（〜切り日もしくは〜切り時間迄存在する権利）によつて条件を付けられなくてはならず、又呼び起される回数によつて条件を付けることができる（たとえば販売者は保護されたアプリケーションを１０回実行する権利をユーザに販売することができる）。以下説明するように実行権は実行権のソース（ソフトウエアの販売者）を満足するようにコプロセッサによつて測定できる限り任意の他のパラメータによつて条件を付けることができる。条件を付された実行権が使用されると、ソフトウエアの販売者にさらに柔軟性が与えられ、ソフトウエアの分野に始めて真に安全な「返品」対策の可能性が開かれる。明らかに、現在のソフトウエアの頒布技術を使用するソフトウエアの販売者は完全購買信用取引きのソフトウエアの返品を受取るようなことがあれば彼の製品を只で与えるという危険がある。現在の頒布技術では、販売者はユーザがすでにソフトウエアを複製して返品した時に、ユーザが完全に使用可能なアプリケーションのコピーを保持しているかどうかを検証する手段がない。本明細書の原理を使用すると、ソフトウエアの販売者は「返品」対策を具体化でき、ユーザがソフトウエアを返品すると、ユーザはもはや実行可能なコピーを保持できなくなる。

ソフトウエアのコピー防止機構は現実に、現実の装置により動作するためと、別個の実行権がコプロセッサの永久メモリ中に暗号キーの形で存在するので、実行権を記憶するコプロセッサが故障する可能性も考慮しなければならない。このような故障が生じてもユーザの実行権が完全に失われるようなことがあつてはならない。本発明はユーザのコプロセッサが故障した場合における実行権を獲得するかを説明しているだけであり）実行権の処遇については向けられていず、実行権に条件を付ける可能性についての説明がない。

C　発明が解決しようとする問題点

本発明の目的はコプロセッサ中に存在するソフトウエアの実行権に条件を付け、その処遇を決め、もしくはこれを転送することにある。

本発明の他の目的は、実行権を安全に転送する機能を与えることにある。

D　問題点を解決するための手段

ソフトウエアの実行権は他のコプロセッサに転送できもしくは、外部に記憶するためにコプロセッサの外部に移すことができる。いずれの場合にも、実行権を転送する過程は擬似的もしくは複製の実行権を発生したり、許したりするものでないことが重要である。このようなことは勿論コピー防止の目的に反する。以下説明するように、実行権の転送は転送セット（多くの点で実行権を獲得

の消失に対してユーザを保護する装置及び方法を与える。実行権を移動即ち転送する場合と同じように、任意のハードウエアのバックアップ技術（コプロセッサの故障の場合に利用可能になる）は擬似実行権を許容する性質を持つていてはならない。

本発明のハードウエア・バックアップ方法は実行権を不当に複製する機会を最小にし、その活動を妨げる。

条件の付された実行権

実行権に条件を付すためには、上記米国特許出願第９２７６２９号に開示されているシステムに、次のものがなくてはならない。

(1) アプリケーション・ソフトウエアの実行が完全に許される（もしくは許されない）条件のステートメント

(2) この条件を測定できるある客観的規準

(3) 条件と規準を比較して、比較の結果によつて実行されるソフトウエア・プログラム

これ等の目的はユーザもしくはソフトウエアの販売者によつて特別に許可されていない他人がこの条件もしくはこの条件の満足をテストする客観的規準を変えようとする試みに対して安全な方法で満足されなければならない。

本発明に従い、この規準はソフトウエア中に、より具体的にはアプリケーション・ソフトウエアの保護された、即ち暗号化部分中に記述される。上記米国特許出願第９２７６２９号に説明されているように保護されたアプリケーション・ソフトウエアがユーザに使用される唯一の形式は暗号形式においてである。それはユーザはデータ・オブジエクトとして暗号キーにアクセスできず、保護されたソフトウエアを修正したり、読取りすらできないからである。保護ソフトウエア内に実行権の条件を組込むことによつて、ソフトウエアの販売者によつて許可されない限りこれ等の条件はユーザもしくは他人による変更から安全に保護される。プログラムされた規準に対してテストする条件を保管するためには、コプロセッサ中の持久メモリ中のある記憶スペースを使用する。この記憶スペースはすでに暗号化ソフトウエアを解読するのに必要な解読キーを記憶する機能が割当てられている。従つてソフトウエアの特定の保護された部分に割当てられている記憶スペースを拡張して規準とテストされる条件を含めるようにする。メモリが持久性であるために実行権がコプロセッサ中に利用可能であるかぎり、この客観的条件も利用可能である。コプロセッサはその物理的に安全な境界内に連続的に電力が供給される実時間クロックを含むので、時間を含む規準を使用したい時は、時間情報が利用可能である。この情報はコプロセッサの持久メモリ中に記憶され、特定のアプリケーションに割当てられたこのメモリの部分だけがそのアプリケーションによつてアクセスできるので、情報はユーザによる修正から安全に保護される。アプリケーション・ソフトウエアは持久メモリのこの部分に記憶された条件を変更できるかもしれないが、実時間クロックの値は変更できない。

たとえば、ソフトウエアはこのメモリ中に保持している回数を変更することによつて回数もしくは総時間がカウントでき、実行の回数もしくは総時間に関連する規準がもはや記憶した条件を満足しなくなる迄実行できる。

たとえば、ソフトウエアの販売者がユーザにある期限の日付を超えてはならないという条件付き（たとえば、ユーザは１９８７年の５月１日迄は保護されたアプリケーションを実行できるが、その後は不可という権利を有する場合）で実行権を転送したものと仮定する。従つてコプロセッサの動作命令はソフトウエア解読キーとともに最後の許可日付（期限の日付）が記憶できなくてはならない。コプロセッサは実時間クロックを保持しているので、解読キーをアクセスする時もしくはアプリケーションの実行の途中にはいつでも、期限の日付及び現在の日付が利用できる。この期限の日付けは実時間クロックのセット時間と同じようにコプロセッサの安全性によつて不法な変更から守られている。ソフトウエアの暗号化部分（保護部分）は実行が期限の日付けを超えていると使用

できないという規準を記述している。保護ソフトウエアが実行される時はいつでも、解読キーと期限の日付けがコプロセッサの持久メモリからアクセスされる。保護ソフトウエア中でテストされる規準は期限の日付を現在の日付と比較することを必要とする。もし現在の日付が期限の日付を越えていると、保護ソフトウエアの実行は進行しない。保護ソフトウエアは又現在の日付が期限の日付を越えると特定のソフトウエア解読キーを抹消できるように作成できる。この分野の専門家にとつては期限の日付に代る条件としてソフトウエアを実行する回数を使用できることは明らかであろう。この場合、保護ソフトウエアはソフトウエア解読キーとともに現在のデータを記憶する代りに、許可された実行回数を記述し、許容使用回数を記憶していて、この回数をソフトウエアの実行のたびにデクレメントする。従つて保護部分が許容実行回数を規準と比較して、残りの回数が0より大きいことを確かめる。許容実行回数が0になると、ユーザのソフトウエア実行要求は拒否される（おそらくソフトウエア解読キーも抹消される）。このような特定の具体例には経過時間、パスワード及びこれ等の組合せ並びに他の測定可能な変数を含む多くの変形が存在することは明らかであろう。

実行権の転送

1人のユーザから他のユーザへの実行権の転送（具体的にはソース・コプロセッサからシンク・コプロセッサへの転送）は頒布セットを再構成することによつて達成できる。この手順は最初に実行権を獲得した形と実質的に同じであるポータブルな形に戻す（上記米国特許出願第927629号参照）。この手順は必然的に実行権をソース・コプロセッサから移動させる。

このトランザクションはユーザがトークン（一種の割印）の構造に依存してトークンもしくはディスクのいずれか及びトークン対（転送セットとも呼ばれる）を得ることを必要とする。これ等のセットはハードウエアの販売者によつて与えることができる。これ等のセット中のトークン（即ちカートリッジ）はコプロセッサのハードウエア製造者によつてロードされる。ユーザによつて使用される前の転送セットは2つの形で記憶される単一の情報片であるトークン・データを有する。トークンはハードウエアの販売者によつて平文のトークン・データとしてロードされる。トークンの物理的特性がこの高度に機密性の情報を不許可の者から保護する。同じデータが共通スーパーバイザ・キー（CSK）と呼ばれるハードウエアの製造者の秘密のキーによつて暗号化され$E_{CSK}$（トークン・データ）が発生される。これは転送セットのディスク上もしくは（可能な構造になつていれば）トークン中に記憶される。$E_{CSK}$（トークン・データ）は暗号化されているから、誰もが読取ることができ、さらにコピーできるようなディスク上に記憶できる。転送セットはハードウエアの販売者のような信頼の置けるソースによつて製造されることが必要である。それは仮にトークンの内容が知られると、他のトークンにその知られた内容がロードでき、転送される実行権が複製されるからである。ユーザが適切な転送セットを獲得したと仮定すると、頒布セットはたとえば次のようにしてユーザと彼の複合計算システムによつて、頒布セット再構成（RDS）過程を使用して作成される。

上位計算機上で実行中のユーテイリテイ・プログラムが（ソース）コプロセッサにRDSシーケンスが開始することを知らせる。ユーテイリテイ・プログラムはコプロセッサに転送すべきキーの位置を知らせる。コプロセッサはインデックス付きキーを除くすべての許可されたキーについてCBS（バックアップ・セット作成）手順を実行する。CBS手順については後に説明する。この時点では、CBS手順は存在する任意のハードウエア・バックアップ機構を無効にすると述べておくだけで十分である。コプロセッサは転送セットから暗号化されたトークン記述子$E_{CSK}$（トークン・データ）のコピーを要求してこれを受取る。コプロセッサはトークン記述子を解読して平文のトークン・データを与える。この平文トークン・データは次にインデックスによつて識別されるソ

フトウエア解読キーを使用して暗号化され、$E_{AK}$（トークン・データ）を発生する。次にコプロセッサはこの暗号化トークン記述子$E_{AK}$（トークン・データ）をトークンもしくはディスク上の予約された持久メモリ領域中に記憶し、所定のメモリ位置にあるソフトウエア解読キーＡＫを抹消即ち非活性化する。従つてコプロセッサは暗号化トークン記述子を上位計算機に渡して転送セット・ディスク上に記憶する。以下説明するように、転送すべきキー（ＡＫ）は実行の条件に関連付けられる。もしこれ等の条件が（期限の日付のように）不変ならば、暗号化アプリケーション・キーは転送セット・ディスクにコピーできる。もし実行の条件が変ると（使用の残り時間もしくは使用の残り回数のように）、アプリケーション・キーを含む暗号化ファイル及び実行の条件はこの条件をリセットすることなくしては頒布ディスクからのコピーができない。トークン記述子ファイルとアプリケーション・キー・ファイルのこの同期は各ファイル中に照応テスト回数を含ませることによつてクアップ・コピーからコピーできる。頒布セットは一つの点を除いてすべての点で同じであるから、これは任意のコプロセッサによつて特定の暗号化ソフトウエア・ファイルを実行する権利を転送するように使用可能である。この頒布セットと実行権を獲得した原頒布セット間の唯一の差異はトークン・データが異なつている可能性が極めて大きいことである。しかしながらこのことはセットの両方の部分（カートリッジ及びディスケット）が同一のトークン・データから得られる限り重要ではない。（ソース）コプロセッサはここで一時メモリから解読キーＡＫを抹消する。この時点で実行権はソース・コプロセッサから抹消され、転送セットだけに存在する。

しかしながら、実行権の転送は間接的、たとえば上述のように転送セットを介して行う必要はない。実行権の転送は直接的でよく、コプロセッサからコプロセッサへの通信リンクを使用しても行うことができる。さらにコプロセッサ間の通信リンクは転送される実行権の安全を守るために保頒て達成できる。転送の次の段階で転送ディスク上に記憶させるために暗号化アプリケーション・ファイルを作成する。この作成は照応テスト回数が乱数に代る点を除き以下に説明する販売者キー暗号化（ＥＶＫ）と同じである。この照応数はトークン・データの一部であつてよい。この作成と転送の後に、上位計算機中で実行されているユーティリティ・プログラムは転送セット・ディスクに保頒されたプログラムの平文と暗号文の部分を含む２つのファイルを転送する。ここでこの頒布セットはこれがソフトウエア解読キーＡＫで暗号化された暗号化トークン・データ、平文ファイル及び同じくソフトウエア解読キーＡＫで暗号化された保頒された即ち暗号化ソフトウエア・ファイル並びにハードウエアの販売者のキーＣＳＫによつて暗号化されたソフトウエア解読キーを含む点で上記米国特許第９２７６２９号に開示されている頒布セットと同じことがわかるであろう。後の３つの要素はもし実行条件が許容されているならば原頒布ディスクもしくはこれ等のファイルのバッされている必要はなく、安全は以下説明するように暗号化によつて保頒される。以下の説明ではコプロセッサ間の転送について説明するが、コプロセッサ間の通信リンクは直接でもよく、双方向データ通信システムを介するものでもよい。１つのコプロセッサから他のコプロセッサへの実行権の転送は関与する２つのコプロセッサが互を「ファミリのメンバ」として確認でき、このトランザクションに使用するための一回限りのセッション・キーを発生できるときは安全と考えることができる。ファミリのメンバであることの識別は各々の使用する手順が正しくかみ合い、従つて実行権の拡散が生ぜず、他の保頒された情報が露顕しないことを保証するために必要である。

セッション・キーを発生するトランザクションはこのトランザクションに関与する２つのコプロセッサ・システムに共通な高い特権メモリ中に情報が存在すること、及びコプロセッサが良い乱数を発生することができる能力に依存する。セッション・キーを発生する過程は両コプロセッサが必

要な前提情報を共通に保持している限り、トランザクションに関与する両コプロセッサに同じキーを所有させることができる。

相互識別とセッション・キーの発生のトランザクションは次のように行われる。説明の目的のために、実行権があるユーザ、複合計算システムもしくはコプロセッサから転送される時はこのユーザ、ユーザの複合計算システムもしくはユーザのコプロセッサをソース・ユーザ、ソース複合計算システムもしくはソース・コプロセッサと呼ぶことにする。実行権が転送されて来る時は、そのユーザ、複合計算システムもしくはコプロセッサをシンク・ユーザ、シンク複合計算システムもしくはシンク・コプロセッサと呼ぶことにする。ソース・ユーザはソース・コプロセッサにセッション・キーが必要になつたことを信号する。ソース・コプロセッサはセッション・キーを発生するのに使用する乱数を発生し、これを他のランダムなビット・ストリングに埋込み、メッセージ確認コード（ＭＡＣ）を添付する。ＭＡＣは解読によつて復元した平文メッセージが転送されて来たメッセージに同じであることを保証するのに使用できる。ソース・コプロセッサはこの数をキーＣＳＫで暗号化し、暗号化した数をシンク・コプロセッサに送る。シンク・コプロセッサも同じ機能を遂行してその暗号化乱数をソース・コプロセッサに送る。ソース・コプロセッサは暗号化キー（ＣＳＫ）によつて受取つた乱数を解読する。各コプロセッサが多くのＣＳＫを記憶している場合には各コプロセッサは受取つた数を各ＣＳＫによつて相継いで、有効なＭＡＣが得られるか、スーパーバイザ・キー（ＣＳＫ）のコレクションが尽きる迄解読する。有効なＭＡＣが発見されなかつた時は、誤りメッセージが戻される。この誤りメッセージの発生はファミリのメンバでないプロセッサがそうであるものとしてそれ自身を偽装した場合の代表的な結果である。もし有効なＭＡＣを得た場合には、２つのコプロセッサ中で発生した乱数は両コプロセッサ中で組合され、セッション・キーとして使用される。勿論コプロセッサのための動作命令の一部は乱数を組合して、セッション・キーを発生する特定の方法を前もつて決定している。ここで、１つの乱数は他の乱数と連結でき、もしくは排他的にＯＲできる等々である。

正しいスーパーバイザ・キーの検索において相続く解読を必要とする手順を避けるために、使用したスーパーバイザ・キーのためのインデックスのような位置情報を暗号化した数とともに送ることができる。しかしながら、コプロセッサに正しいスーパーバイザ・キーを探索させることを要求する技術には長所がある。暗号化した数とともにインデックスされた数を送る手順は傍受者に対してスーパーバイザ・キーに関する情報の集まりを増大させる。

セッション・キーが一度発生すると、コプロセッサは実行権をセッション・キーで暗号化して転送することが可能になる。この手順を制御するコプロセッサのファームウエアは実行権がソース・プロセッサから抹消されることを保証する。実行権はソース・プロセッサによつて転送される時に抹消できるが、好ましい技術はシンク・コプロセッサが実行権を安全に受取つた時にのみ実行権（ソフトウエア・キー）を抹消すること及びソース・コプロセッサが実行権が非活性化されたことを示す時にのみ実行権をアクティベートするものである。暗号化されて転送される実行権は、その暗号化によつて傍受者から安全であり、安全な通信リンクもしくはチャネルを使用する必要はない。

直接、即ちコプロセッサ間通信は回路網もしくは本体リンク環境で実行権を移動する方法である。これ等の場合においてもしくは特定のコプロセッサ中の特定のアプリケーション中に２つ以上の権利を設定することを含む他の場合には、所与のパッケージ当りの権利のカウントがソース・コプロセッサのキー記憶領域中に保持され、すでに受取られている権利の数が頒布されて来る権利を制限するようにされる。

上述のことから、ＡＫをいくつかの次元（時間、使用回数等）のＡＫの寿命の記述に関連付けるの

はこの機構の領域内であることは明らかであろう。

この機構の能力にはソフトウエア販売者によつて与えられ、コプロセッサによつて実行されるソフトウエアの制御の下にこの寿命を分割し、区分することが含まれることが明らかであろう。ソフトウエアのライブラリは実行権の保存を侵害することなく、分散計算システムのユーザに利益を与えるこれ等の手段によつて分散計算システム中に保持できる。

複数の実行権の転送を必要とする場合には、コプロセッサ間の転送は次のように一度セッション・キーが作成されてから行われる。ソース・ユーザはソース・コプロセッサに対しこれ等の実行権（ＡＫ）を転送させることを確認させる。ソース・ユーザはＣＢＳ手順（以下定義する）を実行し、実行権のバックアップ・セットを更新する。次にソース・コプロセッサはセッション・キーによつてソフトウエア・キーを暗号化し、これ等をセッション・キーとともに一時メモリの予約された位置中に記憶する。ソース・コプロセッサは各ソフトウエア・キーをキー・メモリ（永久メモリ）に非活性としてマークし、暗号化情報をシンク・コプロセッサに送る。シンク・コプロセッサは暗号化キーの集りをソース・コプロセッサから受取り、これ等をセッション・キーによつて解読する。シンク・コプロセッサは次にキー・メモリ（永久メモリ）に解読したキーを記憶し、これ等を非活性とマークし、その上位計算機に位置情報を送る。この位置情報によつてアプリケーションはロード解読実行手順でキーをアクセスできる。実際に保護されたソフトウエアより成るファイルは通常の（非保護の）技術もしくは装置によつて送ることができる。シンク・コプロセッサはソース・プロセッサにメッセージを戻して特定のソフトウエア・キー（単、複）の受取りを示して、ソース・コプロセッサにその一時メモリからこれ等のソフトウエア・キーを除去せしめる。ソース・コプロセッサがこの除去を確認すると、キーがシンク・コプロセッサによつて活性化される。

ソフトウエア販売者は実行権の転送が可能なのを欲しないかも知れず、それによつて販売の条項もしくは条件が定められることに注意されたい。実行権の許容可能な移動を制御する条件はＡＫを獲得した手順の１部によつてＡＫと関連して記憶できる。

ＡＫの処遇がユーザによつて要求されると、これ等の条件（フラグ）は、要求された処遇を具体化したコプロセッサ・ファームウエア中に記憶された遂行／非遂行規準とテストされる。

実行権のバックアップ

以下説明するバックアップ手順はユーザのコプロセッサ中に記憶した実行権の集りが、不測の故障、たとえば、コプロセッサの持久メモリの電源の故障によつて失われないことを保証するものである。このＡＫのバックアップ手順はデータ及びソフトウエアのファイルに適用されるバックアップ・手順とは区別すべきものである。後者の手順は完全に通常のもので、一般に知られているものであり、このシステム中に存在する時は、これ等の種類のオブジェクト（平文及び暗号化したソフトウエア並びに暗号化したアプリケーション・キー）に適用できるものである。これ等のオブジェクトの多くの機能的コピーは任意の認可システムによつて使用するために保存され、損失に対処できるようになつている。これに対し、以下説明するバックアップ手順の目的は実行権をバックアップして、これ等が失われることなく、この種のオブジェクトの多くの機能的コピーが作成できないようにすることにある。

このバックアップ手順はコプロセッサ中の不測の故障の影響を免がれるように設計されているので、このバックアップ手順はプロセッサの外部で十分な情報を作成できて、完全に有効な実行権を記憶する（ソース）プロセッサとは完全に独立に実行権設置できるものでなくてはならない。このような独立性のために、バックアップ手順自体が（複製もしくは擬似的の）実行権の可能なソースになる。バックアップ権利の作成の許可はソフトウエア販売者の施策決定による。このバックアップ権利の作成はソフトウエアの販売者に損害を与

える可能性があるが、カスタマの便宜のために与えられるものである。権利の転送に関してすでに説明したように、ソフトウエアの供給者はバックアップを許可することを望まないであろう。この販売の条件はＡＫに関連する非バックアップ・フラグが存在すると、転送の制御について説明したのと同じ機構によつて強化できる。このオプションは実行権が変更できるような場合に望ましい。

バックアップ権の影響を最小にするために、これ等はこのシステム中では条件付きにされ、即ち実際に故障したコプロセッサを有するユーザが例えばハードウエアの製造者によつてこの故障を検査させるに十分長い、適当な短かい時間間隔の後に切れるようにされる。ハードウエア製造者がユーザのコプロセッサの故障を検査したと仮定すると、ユーザにバックアップ権に与えられた条件を除去する追加の手順を遂行する手段が与えられる。バックアップ権の発生と、以下説明する他の能力とを安全に統合するために、バックアップ権の発生は最初はこの時未完のまま残されているすべてのに、設置された実行権の縮小したセットを正しく反映したバックアップ・セットを転送後に直ちに形成することもできる。

バックアップ権を発生するための手順は単一のバックアップ・セットが多くのアプリケーションのために潜在実行権を保持できる点で包括的である。この手順はユーザの実行権の集りが動的であること、たとえばある日の実行権の集りが他の日の実行権の集りと比較して大きいかもしくは小さいことを考慮している。これによつてバックアップ権も動的であり、ユーザの実行権の集りの変更を追跡することが要求される。

基本的には、このバックアップ手順は２段階過程であり、第１段階では、バックアップ・セットが始めて発生される（バックアップ・セット作成、即ちＣＢＳ）。バックアップ・セットは（一群のアプリケーションのための）実行権の暗号化ファイルを記憶するディスクを含む。この（ソース）プロセッサに一意的なスーパーバイザ・キー（ＵＳＫ）も又このファイル中に含まれる。この情報は許容された権利の転送であると考えるべきである。転送手順は必然的にバックアップ手順と関連付けられ（可能な実行権を保持している）ユーザのバックアップ・セットは完全な転送が行われる時に無効にされる。これによつて転送された権利が同じように有効なバックアップ中に存在するのが防止される。従つて、ユーザはバックアップ手続と関連して次のオプションを有する。

(1) ユーザの実行権がコプロセッサ上に設置される時は、ユーザは実際の、但し条件の付された実行権に変換できるバックアップ・セット（潜在的な実行権の集りを表わす）を持つことができるか、

(2) ユーザは１乃至それ以上の実行権を彼のコプロセッサから移動してこれ等を転送セット中に設置するか、もしくは直接他のコプロセッサに転送できるが、この手順は現存のバックアップ・セットの無効化を必要とする。この条件はユーザの権利と完全に両立する。それは実行権がコプロセッサから除去される時は、権利はバックアップ・セットに表わされる必要がないからである。明らかはハードウエアの製造者が故障の証拠として受取つた時に、故障したプロセッサがこれ等の権利を含むことを主張できるプロセッサであることを検査させるものである。この集りをコプロセッサに対して有効にするトークンも又このセットの一部である。このファイルを暗号化するのに使用できるキーはコプロセッサによつて、たぶん日付及び時間を一意的なスーパーバイザ・キー（ＵＳＫ）によつて暗号化することによつて発生される乱数である。バックアップ・セットのトークンによつて有効にされるのはこの乱数キー（ＲＫ）の使用に対する許可である。上述の如く、バックアップ・セットは保留中の転送と考えることができる。

バックアップ過程の第２の段階はバックアップ・セット設置（ＩＢＳ）である。この段階ではトークン及びディスクより成るバックアップ・セットによつて表わされる潜在実行権がコプロセッサ上に設置される。ユーザはＩＢＳ段階を実行する前の複数の機会にバックアップ手順のＣＢＳ段階を実行する可能性があると考えているので、バック

アップ・セットに使用されるトークンは頒布セットに使用するトークンよりも能力がなければならない。この能力は新らしいアプリケーションが獲得されるたび毎にＣＢＳ段階を実行するという事実のために必要である。ユーザの現存のバックアップはこの時無効にされて、追加的な実行権の発生を防止しなければならない。従つてこのトークン・データは代表的には頒布セットに使用できるトークン・データよりもはるかに長く、従つてこのトークンの１部が各ＣＢＳトランザクションの１部として読取られ、この結果トークンの内容が変化し、トークンを完全に放出し尽す過程においてではなく途中で、前のバックアップ・ディスク・ファイルが無効になる。同じトークンがもし必要ならばＩＢＳトランザクションに使用できる。現在はこのようなバックアップ・セットのトークン上にシフト・レジスタ・メモリを使用せず、メモリはランダム・アクセス・メモリとして設計できることを考慮している。しかしながら以下の説明では簡単のために、上記米国特許第９２７６２９号と同じく、シフト・レジスタに基づくが、余分の長さのシフト・レジスタを含むものと仮定する。

ＣＢＳ段階を開始する前に、ユーザはトークンとディスクより成るバックアップ・セットを入手している。ユーザが獲得したディスクは暗号化（された）トークン・データ$E_{CSK}$（トークン・データ）を記憶している。トークンは獲得した時は平文のトークン・データを記憶している。

ＣＢＳ段階は次のように進行する。

ユーティリティ・プログラムがコプロセッサにＣＢＳシーケンスが開始することを知らせる。

コプロセッサはバックアップ・セットのトークン部分に対応する暗号化トークン・データを要求して、これを受取る。

コプロセッサはトークン・データを解読し、使用される分のトークン・データを選択する。トークン・データの一部の使用とはトークン・データのこの部分が破壊されることを意味する。トークン・データの残りの部分はＣＢＳ手順のその後の実行、たとえばユーザがその後のアプリケーションを得て、彼がこれをバックアップする権利を有する時に使用される。

コプロセッサは使用すべきトークン・データの長さに等しい長さの乱数を発生し、この乱数を使用してトークンを読取る、即ち照合して、トークンの応答を得る。コプロセッサはすでに平文のトークン・データ及びこれが発生した乱数が利用できるので正しい応答を予測即ち計算できる。トークンの計算された応答と実際の応答が比較される。もし比較がトークンの実際の応答と計算した応答が一致しないことを示すと、無効のトークンが提示されたことになり、誤りメッセージが戻され、このシーケンスは終る。しかしながら計算した応答と実際の応答が一致すると、トランザクションは継続する。バックアップ・トークンを読取る、即ち照合することによつて、その中に含まれるトークン・データが変更される（検査されるだけでなく）。トークン・データの変更によつて前のバックアップのすべてのコピーを無効にする。従つてトークンに必要とされる長さについて云えば、トークンのビット長をＣＢＳ段階が実行される各度に使用される部分のビット長で割つた値の回数使用されるに過ぎない。トークンがもう１回トランザクションを行うのに十分なだけのデータを含む時は、このデータが読取られて破壊され、新らしいバックアップ・トークンを開始できる。

このトランザクションの残りについては、コプロセッサ中で使用する暗号システムは任意の数字を有効なキーにすることができるものと仮定している。従つて乱数は有効なキーである。この性質はＤＥＳシステムの特徴である。

トークンの確認を検査した後、コプロセッサは第２の乱数でトークン・データの残りの未使用の部分、そのキー・メモリの許容部分及びそのソース・プロセッサを識別する一意的なスーパーバイザ・キー（ＵＳＫ）を暗号化する。上述のように暗号化されたブロックは次にディスク上に記憶できる。このファイルはバックアップ設置手順のために使用できる。

上述の暗号化キーとして使用できる乱数のコピーはここで暗号化販売者キー手順を使用してスーパーバイザ・キーによつて暗号化される。このデータはここでバックアップ・ディスク上に記憶できる。

これ等のファイルは保護されるソフトウエアがコピーできるのと同じように、無限回コピーできる。しかしながら、このファイルはトークンとともにだけ使用でき、従つてトークンが途中で読取られない場合だけ使用できる。すでに説明したように、その後の転送動作はトークンを無効化できる。ユーザがＣＢＳ段階の最後の実行中に発生されたファイルの両方及び中間で読取られていないトークンにアクセスしたものと仮定して、バックアップ・セット設置段階を説明する。

さらに、中間に、ユーザのコプロセッサ、即ちソース・コプロセッサが故障したが、ユーザは他の（シンク）コプロセッサが利用できるものと仮定する。バックアップ・セット設置手順はシンク・コプロセッサ上にユーザのソース・コプロセッサ上に（バックアップの許可される程度に）存在した実行権のすべてを設置できる。しかしながら、ソフトウエアの販売者を保護するために実行権の全集りが許可期間によつて条件付けられる。許可期間中に、実行権の各々は動作できるので、ユーザは彼の（ソース）コプロセッサ中でこれ等を使用できる。許可期間中に、ユーザは彼の故障した（ソース）コプロセッサをハードウエア販売者に戻すことができる。ハードウエアの販売者は、コプロセッサが実際に故障したことを検査した後、（シンク）コプロセッサに（ソース）コプロセッサが故障しているので実行権についての許可期間の条件を引上げることができることを確認させる暗号化したメッセージを有するディスクをユーザに与える。従つてＩＢＳ段階は２つのサブ段階を含む。第１のサブ段階は実行権を（シンク）コプロセッサに設置する。これ等の実行権を設置する時に、これ等は条件が付されて、許可期間が切れると無効になることを示す。ＩＢＳ段階の第２のサブ段階は実行権についての条件をハードウエア販売者から獲得したディスクを使用することによつて除去する。

次にＩＢＳ段階は次のように進行する。

ユーティリティ・プログラムがシンク・コプロセッサにＩＢＳシーケンスが始まろうとしていることを知らせる。最初コプロセッサにはバックアップ・ファイルを暗号化するのに使用する暗号化乱数キーを要求して、これが送られてくる。このキーは適切なＣＳＫによつて解読される。次にコプロセッサは暗号化バックアップ・ファイルを要求して、それが与えられる。暗号化バックアップ・ファイルは暗号化したトークン・データ、（ソース）コプロセッサからのキー・メモリの内容及びソース・コプロセッサの一意的なスーパーバイザ・キーを含む。

（シンク）コプロセッサは前の段階で見出した乱数キーを使用してバックアップ・ファイルを解読し、トークン・データを取出し、トークンを検査する。トークン検査は頒布セットの検査と同じく進行する。即ち検査はトークンの内容を変更する。この結果、トークンはＩＢＳ手順を一回だけ遂行するのに使用できる。トークンが有効であると仮定すると、バックアップ実行権のセットが（シンク）コプロセッサの永久メモリ中に許可期間の範囲を決定する〆切りの日付とともに設置される。

ここで（シンク）コプロセッサによつて両コプロセッサの一意的なキーのコピーを含むメッセージが準備される。このメッセージは故障したソース・コプロセッサとともに戻され、ハードウエアの条件引上げメッセージはシンク・コプロセッサによつてシンク・コプロセッサに向けられたものであること、及び特定のソース・コプロセッサの実行権のセットのためのものであることが識別される。

この点で、（シンク）コプロセッサはすべての実行権が〆切りの日付について条件付けられた点を除き、（ソース）コプロセッサが最後にバックアップされた時と同じ条件にある。ユーザがハードウエア販売者から許可期間内に検査ディスクを

受取つたと仮定すると、ＩＢＳ段階の第２のサブ段階は次のように進行する。

ユーテイリテイ・プログラムは（シンク）コプロセッサにＩＢＳシーケンスが完了しようとしていることを知らせる。（シンク）コプロセッサは検証ファイルを要求し、これを受取つてこれをスーパーバイザ・キー（シンクのＵＳＫ）によつて解読する。もしファイルが（ソース）コプロセッサを識別する一意的なキーを含むならば、条件が引上げられる。もしファイルが（ソース）コプロセッサの一意的キーを含まない時は、誤りメッセージが戻され、サブ段階は完了する。

プロセッサ間通信によるバックアップ

前の章では、ハードウエア・バックアップはともに拡張トークンに依存するＣＢＳ及びＩＢＳ手順より成るものとしてハードウエア・バックアップを説明した。しかしながら、トークンもしくは拡張トークンの使用はハードウエア・バックアップにとつては不可欠でなく、中間のコプロセッサが先ずソース・コプロセッサと（ＣＢＳ手順のために）通信し、次にシンク・コプロセッサと（ＩＢＳ手順のために）通信できるように配列できる限り、ハードウエア・バックアップの目的のために、トークンもしくは拡張トークンに代つて中間のコプロセッサが使用できる。

ハードウエア・バックアップ手順のプロセッサ間型の変形について説明する。

この説明の目的のために、ソース・コプロセッサは中間のコプロセッサと通信するものと仮定する。コプロセッサは最初送信するコプロセッサが選択したスーパーバイザ・キーによつて各々暗号化した暗号化乱数を交換する。既に説明したように、コプロセッサの各々は最初は使用された特定のスーパーバイザ・キーを知らないにもかかわらず、他のコプロセッサによつて転送されてきた暗号化乱数を解読して認識できる。受信するコプロセッサの各々は暗号化乱数を成功裡に解読したと仮定すると、これ等はこれ等の平文乱数を組合してセッション・キーを発生し、その後の通信に使用する。

次にソース・コプロセッサはその永久メモリからアプリケーション・キー（ＡＫ）即ちバックアップされることが求められているキーを引出し、このキー情報及びそのＵＳＫをセッション・キーによつて暗号化して暗号化キー・ブロックを発生する。これはバックアップ・デイスク上に記憶される。セッション・キーは中間のコプロセッサによつて、このキーがバックアップ・セットに対応することを示す記述子とともにその安全なメモリ中に記憶される。この時点で、中間のコプロセッサはバックアップされつつあるＡＫ（単もしくは複数）にアクセスするのに必要なキーを所有するが、このコプロセッサが信頼性のある受取り手でない場合は偽の実行権を発生できる。しかしながら、上述の識別化手順によつてソース・コプロセッサには中間コプロセッサが一族のメンバであることが知らされている。そのようなファミリのメンバとして、中間のコプロセッサの論理及び物理的安全性が転送されて来た実行権を偽の複製もしくは使用から防止する。実際の暗号化アプリケーションは通常の手段によつてＩＢＳ手順のために中間のコプロセッサに送ることができる。

もしその後ソース・コプロセッサに中間のコプロセッサを介してバックアップされている実行権を転送するように要求されると、上述のようにソース・コプロセッサが中間のコプロセッサと通信してバックアップ権を無効にする迄はトランザクションは生じない。このような通信は３つの主要部分を有するだけである。第１の部分はすでに説明された識別子シーケンスである。これによつてソース及び中間のコプロセッサはこの通信の当事者が実際に一族のメンバであるという条件が満足される。第２の部分はソース・コプロセッサからの中間のコプロセッサへの、バックアップ権のキーの無効を要求するメッセージの転送である。第３の部分は、自分がバックアップ権を記憶する真の中間のコプロセッサであり、権利が無効にされたことを示す中間のコプロセッサからの返答である。中間のコプロセッサ中に記憶されているバックアップ権はＩＢＳ手順を使用して中間のコプロ

セッサ中に設置できるか、もしくは異なる（シンク）コプロセッサ中に設置される。バックアップ権が中間のコプロセッサ中に設置される場合には、この動作は中間のコプロセッサに関連する上位計算機を介してユーザによつて容易に行われる。この権利はトークンを必要とすることなく設置できるが、すでに説明したように、これ等の権利は許可期間によつて条件が付されている。

中間のコプロセッサを使用するＩＢＳ手順が遂行される場合には、両コプロセッサは先ずセッション・キーを確立することによつてファミリのメンバであるとして相互のアイデンティティを確立する。次に乱数キーがセッション・キーによつて暗号化され、シンク・コプロセッサに転送できる。シンク・コプロセッサがこのようにして許可され、許可期間を有するバックアップ実行権のセットが設置でき、上述のようにハードウエアの製造者のためのメッセージが準備される。

販売者のキーの暗号化

本明細書及び上記米国特許第９２７６２９号に説明されているように、コプロセッサは又ソフトウエア解読キー（ＡＫ）を暗号化するというサービスをソフトウエアの販売者に与える。全ソフトウエアのコピー防止システムによつて与えられる保護にとつては、ソフトウエアの販売者の解読キーを暗号化するのに使用するスーパーバイザ・キー（ＣＳＫ）が秘密に保持されることが重要である。ソフトウエアの販売者にコプロセッサの使用と無数のキーを暗号化する能力を与えると、ソフトウエアの販売者はハードウエアの販売者の暗号キー（ＣＳＫ）に通常行われている選択された平文（を求める）侵害を試みることができる立場になる。ソフトウエアの販売者がハードウエアの販売者のキー（ＣＳＫ）を知ることができると、そのソフトウエアの販売者はこれ等のキーを使用して他のソフトウエアの販売者の暗号化ソフトウエアを侵害することができる。選択された平文を攻略するには、侵害者が対応する平文及び暗号テキストのセットにアクセスできることが必要である。これ等セットを使用して、侵害者は平文から暗号テキストを作成した、もしくは逆の解読に使用するキーを識別しようと試みる。この試みの困難さは、もちろん、選択した暗号システム及びバックトラック・タスクに利用できる計算能力による。ある暗号システム（ＤＥＳのような）は非限定回数試みられる平文攻略に極めて抵抗性がある。

平文の攻略に対するコプロセッサの使用についての第１の制約では、コプロセッサはソフトウエアの販売者がハードウエアの製造業者からの許可を求めて実行権を発生するようになつている。この許可はハードウエアの販売者によつてソフトウエアの販売者に販売されるようになつている。従つて、平文の攻略に対するコプロセッサのソフトウエアの販売者についての１つの制約は、平文及び対応する暗号テキストのセットの発生に支払われる経済的コストである。他の技術は次の販売者キー（ＥＶＫ）手順の使用である。

ユーティリティ・プログラムはコプロセッサにＥＶＫシーケンスが開始しようとしていることを知らせる。コプロセッサは暗号化すべきキー（ＡＫ）、代表的にはソフトウエア販売者のキーを要求して、これを受取る。キーはソフトウエアの販売者によつてスーパーバイザ・フラグの所望のセット値とともに与えられる。これ等のフラグはスーパーバイザが（たとえば）関連キーのバックアップもしくは転送を許容するかどうかを制御するものである。

アプリケーション・キーによつて検査されるか変更される（あるいはこの両方を行う）実行の条件も与えられる。次の説明でＡＫはこの過程による転送に準備された全データ（フラグ、キー及び条件）をさすものとする。

コプロセッサは乱数（ＲＮ）を発生しこれをキーＡＫの前端に埋込む。コプロセッサはキーの後端に認識フラッグ（ＲＦ）を埋込む。上述のように、ＲＦは正しいＣＳＫが（解読中）に使用されたことを検査し、コプロセッサによつて正しい暗号システムが使用されたことを検査するのに適した任意のＭＡＣでよい。乱数及び認識フラグはユーザ、たとえばソフトウエアの販売者には知られ

ていない。コプロセッサは結果のブロックをスーパーバイザ・キー（ＣＳＫ）によつて暗号化し、この結果をユーティリティ・プログラムに渡す。この結果、$E_{CSK}$（ＲＮ，ＡＫ，ＲＦ）はＣＳＫを知つている任意のコプロセッサによつて解読できる。ここで「．」はストリングの連結を示す。即ち０１．１１１はストリング０１１１１を生ずる。解読の結果のＲＮ、ＡＫ、ＲＦは３つの要素、乱数（ＲＮ）、解読キー（ＡＫ）及び認識フラッグ（ＲＦ）を含む。解読コプロセッサはサフィックスの認識フラッグの長さに先ずアクセスできる限り常にＡＫを識別できる。乱数の埋込みによつて、たとえ同じキー（ＡＫ）が多数回提示されたとしても、結果のブロック及び暗号化の結果が異なるので、平文の攻略を妨害することができる。ＥＶＫ手順は解読コプロセッサが認識フラッグ（ＲＦ）を先に知つている限り、又すべての可能なＣＳＫの先験知識を有する限り、特定のＣＳＫの先見知識がなくても暗号化ブロックを解読できるという他の利点を有する。さらに具体的に、各コプロセッサにはハードウエアの販売者によつてＣＳＫ１－ＣＳＫ５が与えられているものとする。もし　号侵害を避けるための適切な手段をさらに与えたければ、使用される認識フラッグ（ＲＦ）は常に暗号キー自体であると仮定することができる。これ等の手段はブロック暗号化の場合はＲＮがブロックの長さの整数倍でないものとする。さらに暗号化に使用するコプロセッサは特定のソフトウエア・キーＡＫを暗号化するのにランダムにＣＳＫ３を選択したものとする。従つて暗号化ブロックは$E_{CSK3}$（ＲＮ，ＡＫ，ＣＳＫ３）となる。ＣＳＫ１－５にアクセスできる任意の他のコプロセッサはたとえキーＣＳＫ１－５のうちのどれが使用されたかを知らなくてもこのブロックを正しく解読できる。コプロセッサは唯キーＣＳＫ１－５の各々で暗号化ブロックを解読するだけでよい。唯１つの場合にだけ、平文版はサフィックスとして実際に使用する解読キーを含んでいる。

仮想キー・メモリ

本明細書及び上記米国特許出願第９２７６２９号に説明されているソフトウエア保護機能の実際の具体化には、コプロセッサの実行権を記憶する能力にある制限を与えられなければならない。ユーザがコプロセッサ中に記憶できる数よりも多くの実行権を受取つた場合には、ユーザはめつたに使用しない実行権を転送トランザクションによつて移動可能な形に変換して戻すことができる。これ等の方法によるコプロセッサへのＡＫのスワップ・イン及びスワップ・アウトの動作はユーザの負担になることが容易に明らかであり、このような保護機構の精神に反する。しかしながら、コプロセッサにリソースを与えると、この問題を克服するのは容易である。

最も簡単な場合、ＡＫはソフトウエアの一部のサービスにアクセスできる権利を表わす。この概念をわずかに拡張すると、メタＡＫ（ＭＡＫ）は実行権の集りをアクセスする権利を表わすものとできる。このキーの種類はすでに制限的な形でバックアップ・トランザクションに使用するＲＫもしくはランダム・キーに見られる。

ユーザがそのコプロセッサのすべてのキーの位置を充満し、さらにもう一つを設置しようとしたとすると、コプロセッサはユーザにシステム・ディスク（ハード・ディスクもしくはフロッピー・ディスク）上にキーのファイルを開始させるオプションを与えることができる。この仮想キーの集りはＭＡＫによつて暗号化形で記憶される。この集りがアクセスされてＡＫが移動もしくは加えられる度に、この集りは新らしいランダムなＭＡＫで再暗号化される。ＭＡＫはキー・メモリ中に記憶され、バックアップ及び転送を制御するフラグ中にＭＡＫとしてマークされ、プロセッサによる正しい処理が行われるようにされる。アクセス毎にＭＡＫの変更が必要であり、このようにしてコプロセッサによつて使用される仮想キーがユーザによつて所有される実行権の実際の集りと同期される。転送された仮想実行権を含む旧仮想キー・メモリのコピーはローデングしても正しく解読されないのでコプロセッサ上に紛れ込むことはない。

仮想キーを発生する（キー・メモリを解放する

目的のために）特定の手順は次のように進行する。上位計算機上で実行中のユーティリティ・プログラムがコプロセッサに、ユーザが仮想キーの発生を要求したことを知らせる。コプロセッサはユーザに上位計算機を介して位置情報によつて仮想メモリ内に含まれるべきＡＫを識別することを要求する。ユーザの識別に基づいて、識別されたＡＫはキー・メモリからそのフラグ（転送、バックアップ等のための許可を識別する）、実行条件及び位置情報とともに取出される。コプロセッサは乱数を発生し、そのデータより成るブロックを暗号化する。結果の暗号化ブロックはここでディスク（ハードもしくはフロッピー）に書込まれる。乱数（仮想メモリのためのキー）が次に永久メモリ中に仮想化されたＡＫの１つに代つて書込まれる。仮想キーを乱数のＭＡＫインデックスにマップするキー参照経路ファイルが平文で上位計算機のアクセス・ユーティリティに与えられ、参照のあいまいさが解決される。このファイルは乱数キーを、仮想化したＡＫを識別する位置情報と関連付ける。

デモンストレーション用ソフトウエア

仮想メモリ技術は又トークン・ソースもしくはカートリッジのような検証機構なしでデモンストレーション用ソフトウエアを支援することができる。トランザクションは以下説明するように進行する。キーの転送及びバックアップを許したり許さなかつたりするフラグのセット値はさらにキーの抹消を許したり許さなかつたりするフラグを含む。トークンがない状態でＡＫを受取らなければならない時には、このＡＫをローデングするためのデータ・ファイルは空のトークン記述子を含み、空のトークン・コネクタがトークンの照合に正しく応答する。この特殊なＡＫのためのフラグ・セットはバックアップを可能とするが、移動もしくは消去は可能としないセット値を含んでいる。このようなＡＫの設置には設置されたキー・メモリと仮想キー・メモリの両方を探索して、このキーが前に設置されたものであるかどうかを知る必要がある。

デモンストレーション用ソフトウエアの実行条件には（好ましくは）経過時間もしくは使用回数を含む。上述のような条件の下で進行するＡＫ設置はデモンストレーションの目的のためのソフトウエアの１部の実行権、代表的には１回使用の実行権の獲得を許容する。同じＡＫを設置しようとする２回目の試みは前の獲得を検出する。それはキー・メモリが探索されるが、キーが移動もしくは消去されていないからである。ユーザは無用のデモンストレーション用キーを仮想化することによつて、そのキー・メモリにこれ等の無用のキーで充満させることから守られる。ソフトウエアの販売者はデモンストレーション用ソフトウエアを使用することによつて彼のコードが繰返し、勝手に再設置されることから守られる。

Ｅ　実施例

Ｅ１　ソフトウエア資産保護機構

参照文献として、前記米国特許出願第９２７６２９号及び第９２７２９７号（トークンの実施例）があげられた。米国特許出願第９２７６２９号はユーザが後に仮想キーの１つによつて保護されているソフトウエアの実行を要求する。コプロセッサは先ず参照された位置のキーの使用を試みる。この位置にあるキーの正しさは通常のメッセージ許可技術によつてテストされる。このキーの検査は短かいメッセージ・ファイルに（保護されたプログラムの１部として与えられる）確認セクションをロードして解読することによつて遂行できる。もし解読によつて確認が有効であることがわかると、正しいキーが見出されている。もし確認が無効であると、アクセス・ユーティリティはもしユーザが実際にキーを有する時には、そのキーを含んでいる仮想キー・アクセス経路のリストを与えることができる。これ等のキーは仮想ブロックを解読し、適切なＡＫを引出すことによつて得られる。この時点で、コプロセッサは選択された、前もつて仮想化したＡＫをアクセスしている。

明らかに、仮想キー・メモリはそれ自身にＭＡＫを含むことができ、キー・メモリの寸法は無限に拡張できる。

基本的なソフトウエア資産保護機構を開示している。この機構を簡単に第２図関連して説明する。ソフトウエア資産保護機構はユーザにある通信リンクもしくは経路１４を介して接続した上位計算機１０及びコプロセッサ２０を含む複合計算システムの使用を要求する。米国特許出願第９２７６２９号に説明したように、経路１４はたとえば上位計算機１０とコプロセッサ２０の両方を含むカバー内に封入された内部バスでもよく、コプロセッサ２０内に含まれるＩ／Ｏ装置と上位計算機１０に関連するＩ／Ｏ装置間のリンクでもよい。リンク１４の特定の性質にかかわらず、コプロセッサ２０にはユーザもしくは侵害者によるコプロセッサ２０の内部への機械的干渉もしくはアクセスを防止する効果のある物理的安全性が与えられている。この物理的安全性は内部の破線の境界によつて第２図に示されている。コプロセッサ２０の２つの重要な機能部分は永久（持久）メモリ２５及び一時メモリ２６である。後者は通常の計算機の作業メモリ（ＲＡＭ）と同種のものである。コプロセッサ２０は永久メモリ２５中に少なくとも１つの解読キーＣＳＫが記憶された形でユーザに与えられる。解読キーＣＳＫはコプロセッサ２０の販売者によつて与えられ記憶される。ユーザが保護されたアプリケーションを実行するためには、このアプリケーションを実行する権利を永久メモリ２５中に設置しなくてはならない。この実行権はソフトウエア解読キーＡＫによつて表わされる。上記米国特許第９２７６２９号に説明されているように、ユーザは実行権を設置するために、ソフトウエアの販売者からハードウエア・カートリッジ３０及び頒布ディスク１６を含む頒布セットを受取る。第２図に示したように、頒布ディスク１６は３つのファイルを代表的には記憶している。１つのファイルは保護されたアプリケーションである。代表的な場合、保護アプリケーションは２つのサブ・ファイル中の２つの部分、即ち平文アプリケーション・ファイルＡと暗号化された即ち保護された部分アプリケーション・ファイルＢより成る。第２図に示すように、アプリケーション・ファイルＢはソフトウエア解読キーＡＫによつて暗号化されている。頒布ディスク１６の第２のファイルはキーＣＳＫによつて暗号化されたソフトウエア解読キーＡＫである。最後に頒布ディスク上の最後のファイルはソフトウエア解読キーＡＫによつて暗号化されたトークン・データ$T_1$である。上記米国特許出願第９２７６２９号に開示されているように第３のファイルはかならずしも頒布ディスク１６中に組込まれている必要はなく、ハードウエア・カートリッジ３０中に組込まれてもよい。

ハードウエア・カートリッジ３０は平文形でトークン・データ$T_1$を記憶している。コプロセッサ２０と同じように、ハードウエア・カートリッジ３０は物理的安全性が与えられている。実行権を設置するために、ハードウエア・カートリッジ３０は組合せ計算システムに結合即ちリンクされる。第２図は接続ケーブル１８によるこの結合を表わす。コプロセッサ２０は一時メモリ２６中にある暗号化したソフトウエア解読キーＡＫを解読する。ソフトウエア解読キーＡＫを永久メモリ２５に受入れる前に、コプロセッサ２０はハードウエア・カートリッジ３０が偽造防止性のある照合／応答トランザクションで確認されたものであることが検査される（その内容$T_1$は暗号化ファイル$E_{AK}(T_1)$に対応する）。ハードウエア・カートリッジ３０は読取り後に破壊されるので、まだ使用されていない時はトークン・データ$T_1$を含むだけである。ハードウエア・カートリッジ３０が確認されたものであり、未使用であることを検査し（この過程でハードウエア・カートリッジの有用性を破壊した）後に、コプロセッサは実行権を受取り、その永久メモリ２５中にソフトウエア解読キーＡＫを記憶する。ソフトウエア解読キーＡＫにアクセスすることによつて、保護されたアプリケーション・ファイルＢが解読でき、コプロセッサ２０の一時メモリ２６中に記憶され、コプロセッサ２０によつて実行できるようになる。物理的及び論理的安全性のために、アプリケーション・ファイルＢは実行過程中に平文形で一時メ

モリ２６中に記憶されるが、ユーザもしくは侵害者に利用されることはない。

Ｅ２　実行権の制約

上述のように、ソフトウエア資産保護機構はコプロセッサ２０中に制約のない実行権を設置している。しかしながら本発明の特徴の１つは実行権に条件を付けることである。その例は期限の日付及び時間もしくは実行の回数である。第１図は第２図と似ているが、第１図ではアプリケーション・ファイルの保護部分が実行の規準、たとえば期限の日付もしくは時間が現在の日付けの前ならば実行を許可するという規準を含んでいる。第１図に示したように頒布セットを上位計算機１０及びコプロセッサ２０より成る複合処理システムに提示すると、実行権の設置は正確に第２図に関連して説明したのと同じように、コプロセッサ２０は上位計算機１０の仲介によつてトークン３０の平文の内容（$T_1$）をディスク１６から読取つたファイル$E_{AK}$（$T_1$）の解読版とを比較することによつてトークン３０を検査して（破壊する）。トークン３０の確認及び未使用なことを検査した後、コプロセッサ２０は永久メモリ２５中にソフトウエア解読キーＡＫを読取る。実行の条件はＡＫと同じファイル中に記憶でき、ＡＫと同時に設置される。第１図の場合には、コプロセッサ２０は期限の日付、もしくは時間（あるいはこの両方）として解釈できるデータを記憶している。この解釈は任意の使用時にそのアプリケーションの保護部分によつて遂行できる。期限の日付メモリは第３図に示したようにソフトウエア解読キーＡＫと関連している。具体的に説明すると、第３図はソフトウエア解読キーＡＫ及び期限の日付もしくは時間がコプロセッサ２０の永久メモリ中に記憶されていて、トークン３０が削除されている点を除き第１図と全く同じである。その後、保護アプリケーションがコプロセッサ２０上で実行されるたびに、実行の許可の前に、アプリケーションは暗号化アプリケーション・ファイル中に現在の日付もしくは時間が期限の日付もしくは時間よりも遅くてはならないと述べてある規準を使用し、規準を満足した場合にのみ実行が許可される。現在の日付もしくは時間はコプロセッサ２０による要求によつて実行中のアプリケーションに供給される。

第４図は暗号化アプリケーション・キー・ファイル中に述べられている規準が許可される実行の残り回数を与えている点を除き第３図と類似している。実行権の設置により、ソフトウエア解読キーＡＫはカウントＣと関連し、アプリケーションの実行が要求されるたびに、カウントＣの内容が残された許可実行回数Ｃが０より大きいという規準に対してテストされる。次にカウントＣが１だけ減少される。Ｃが０より大きい限り実行は許可される。

制約が期限の日付であるか時間であるか、もしくは実行の回数であるかにかかわらず、コプロセッサ２０には最初に定めた条件がもはや満足しなくなつた時に関連ソフトウエア解読キーＡＫを抹消する命令を与えることが望ましい。従つてソフトウエア解読キーは永久メモリから自動的に除去され、実行権が削除される。

Ｅ３　実行権の転送

第５図、第６図及び第７図は実行権の転送を説明するためのものである。第５図に示したように、コプロセッサ２０はその永久メモリ２５中にソフトウエア・キーＡＫの形で特定の保護アプリケーションを実行する権利を設置している。ユーザは実行権を転送するために、転送セットを得る。転送セットはトークン４０及び付帯ディスク４６を含む。トークン４０はコプロセッサのハードウエア製造者のような信頼のおけるソースによつてトークン・データ$T_2$がロードされていて、ディスクはハードウエアの販売者のキーによつて暗号化されたトークン・データが書込まれ、従つてディスクはファイル$E_{CSK}$（$T_2$）を記憶している。ユーザは又第５図中に列挙されているファイルを含む原型のソース・デスク１６が利用できる。

第６図は転送シーケンスの第１の段階の後のこれ等の部品の条件を示す。さらに具体的には、転送セットのディスク４６が読取られ、その内容が解読され、コプロセッサ２０がその一時メモリ２

6中にトークン・データ$T_2$を記憶できる。転送されるべきソフト解読キーAKは永久メモリ25から一時メモリ26に移動される。

転送シーケンスの次の段階はキーRKによつてトークン記述子$T_2$を暗号化して$E_{RK}(T_2)$を求めること、ディスク46上に多数のファイル、即ちアプリケーション・ファイルA（平文）、暗号化アプリケーション・ファイル$E_{AK}$（アプリケーション・ファイルB）、ソフトウエア解読キー$E_{CSK}$（AK）（必要ならば上述のようにして準備される）及び暗号化トークン・データ$E_{AK}(T_2)$を書込むことである。従つて第7図に示した、2つの段階の終りには転送セットのディスク46はユーザがすでに取得していた原型のディスク16と略同じものになる。異なる点はディスク46と16上では（暗号形で存在する）トークン・データが異なること、（カウントのような）実行条件が変つた時に、新らしい暗号化アプリケーション・キー・ファイル$E_{CSK}$（AK）が使用されることである。このファイルは上述の同期機構によつて、新らしいトークンに関連付けられる。従つて転送は侵害的手段によつて行なわれると無用のものになる。勿論コプロセッサ20上に実行権を設置する時には、ユーザはトークン・データ$T_1$を含む彼のトークンが使用済であるので（従つて無効化しているので）、原型のディスク16上のファイル・セットを使用して他のコプロセッサ中に実行権を設置できない。しかしながら、ディスク46はユーザが現在所有するトークン40に対応する暗号化トークン・データを含んでいる。従つてユーザは保護されたアプリケーションを他のコプロセッサ上に保護アプリケーションを実行する権利を設置する立場にある。同時に前にソフトウエア解読キーAKを記憶していた元のコプロセッサはもはやこのキーを記憶していないことに注意されたい。従つて保護アプリケーションを実行する権利はコプロセッサ20からカートリッジ40及びディスク46を含む頒布セットによつて表わされたポータブルな形に移されている。

ディスク46及びカートリッジ40を含む転送セットは異なるコプロセッサ上に保護アプリケーションの実行権を設置するのに使用できるが、実行権のための外部メモリの形として利用できる。たとえば、ユーザは望むときに実行権を、最初にこれを引出したコプロセッサ20に戻すことができる。

コプロセッサからコプロセッサへの実行権の転送を第19図及び第1表に示す。第19図はソース複合プロセッサが上位計算機10及びコプロセッサ20を含み、シンク複合プロセッサはシンク上位計算機110及びシンク・コプロセッサ120を含む。第19図に示したように、ソース・コプロセッサは転送さるべき実行権（AK）を含む。一般に両コプロセッサは同じスーパーバイザ・キーの集り（CSK）を含んでいるので、一般化するために、第19図はソース・コプロセッサは以下の過程でスーパーバイザ・キーCSK1を使用し、他方シンク・コプロセッサ120は異なるスーパーバイザ・キーCSK2を使用するものとする。どちらのコプロセッサも他方のコプロセッサがどのスーパーバイザ・キーを使用するかをあらかじめ知つていないものとする。ソース及びシンク・コプロセッサは通信リンク200を介して相互接続されている。

上述のように、一度通信を確立すると、コプロセッサは識別子シーケンスを開始してそれ等自身が他方を一族のメンバであることを確かめる。この過程のあらましを述べた第1表の段階1では、各コプロセッサは乱数を発生し、従つてソース・コプロセッサはRN1をシンク・コプロセッサはRN2を発生する。段階2で乱数の各々は各コプロセッサによつて独立に選択されたスーパーバイザ・キーによつて暗号化され、暗号化した情報を転送して、段階3でシンク・コプロセッサ120は$E_{CSK1}$（RN1）、ソース・コプロセッサ20は$E_{CSK}$（RN2）がアクセスできるようになる。上述のように、ファミリのメンバだけが転送されて来た乱数を解読して認識できる。ソース及びシンク・コプロセッサは一族のメンバと仮定したので、各々は送信されたメッセージを解読

でき、従つてその後両コプロセツサはＲＮ１及びＲＮ２が利用可能になる。すでに説明したように、セツシヨン・キーＳＫは両方の乱数の合成であるから、第１表の段階４で各コプロセツサは独立に同じセツシヨン・キーＳＫを決定できる。

第　１　表

| | ソース・コプロセツサ | | シンク・コプロセツサ |
|---|---|---|---|
| (1) | ＲＮ１ | | ＲＮ２ |
| (2) | $E_{CSK_1}$（ＲＮ１）⟶ | ⟵ | $E_{CSK_2}$（ＲＮ２） |
| (3) | $E_{CSK_2}$（ＲＮ２） | | $E_{CSK_1}$（ＲＮ１） |
| (4) | ＳＫ＝ＲＮ１・ＲＮ２ | | ＲＮ１・ＲＮ２＝ＳＫ |
| (5) | $E_{SK}$（ＡＫ）⟶ | | $E_{SK}$（ＡＫ） |
| (6) | | ⟵ | $E_{SK}$（ＡＫ）の受取証 |
| (7) | ＡＫの抹消 | | |
| (8) | ＡＫを抹消した⟶ | | |
| (9) | | | ＡＫをアクティベート |

その後ユーザが転送さるべき実行権（ＡＫ）を識別した後、ソース・コプロセツサはセツシヨン・キーＳＫによつて実行権を暗号化し、これをシンク・コプロセツサに転送する（段階５）。この時点でシンク・コプロセツサ１２０は実行権ＡＫにアクセスできるが、まだ実効はない。この実行性の欠如はコプロセツサの各々に課せられる信頼のおける安全手順によつて強請されるものである。シンク・コプロセツサはメツセージをソース・コプロセツサに戻し（段階６）、実行権の受取りを示す。これによつてソース・コプロセツサは実行権を抹消できる（段階７）。最後にソース・コプロセツサはメツセージをシンク・コプロセツサに転送し、実行権が抹消されたことを示す。この後初めて、段階９でシンク・コプロセツサは実行権をアクティベートできる。

実行権もしくは実行権の集りの直接転送について説明したので、ネツトワーク中の計算機間のライブラリ型のソフトウエア頒布のための安全な手順を説明することは簡単な拡張である。すでに説明した理由のために、保護ソフトウエアは上述のように各々上位計算機（たとえばＰＣ）及びコプロセツサより成る複合計算システムに頒布できるだけである。たとえネツトワーク中の関与システムの一部が上述のような複合計算システムでない場合でも保護が守られることは明らかである。このような介入システムが存在しても保護されたソフトウエアは簡単にこれ等には利用できない。このようなネツトワークでは、このような複合計算システムの任意の対間ですでに説明した手順によつて実行権を安全に転送できる。転送が観測（及びコピー）されるということはすでに説明した理由で重要でない。実行権の転送を観測する者は往復する暗号化メツセージを単に観測するだけであり、スーパーバイザもしくはセツシヨン・キーにアクセスできないので、安全は任意のメツセージを解読もしくは読取ることができないことによつて保護される。このようなネツトワーク中では、ソースが代表的には単一の保護アプリケーションのための重複実行権を含む多くの実行権を利用できるという意味で１つもしくはそれ以上のソフトウエア・ソースが存在する。明らかにこのようなソースは実行権を多くの他の複合計算システムに転送できる（ソフトウエア・フラグが転送を可能にすると仮定している）。実行権が転送される度に、ソースの権利の集りは転送された権利の分だけ減少し、権利が戻されるたびに、ソースの権利の集りはこの戻りによつて増加する。任意のソースの永久メモリは常に利用可能な権利のカウントを保持し、利用可能な権利の１つもしくはすべてが転送される度に内容が変更できる。ソースはたとえば、被転送ユーザが転送のために払出される口座をもつべきことを規定するように転送される権利を条件づけできることが明らかである。転送される権利は又使用時間もしくは使用回数等によつて条件付けることができる。ある場合、たとえば転送される権利に条件が付せられていて、将来のある日付に時間切れになつている場合には、ソースは前提としている転送トランザクションの日付が過ぎるとその実行権の集りをインクレメントできるようになつている。換言すると、転送できる権利が条件付けられていて、特定の日付けに消滅するようになつているので、ソースにとつては

実際の転送が生じなかつた場合でも、その特定の日付の後に、前に転送した権利を再獲得することは適切である。ソースは又実行回数によつて測られる利用可能な権利のカウントを保持でき、勿論このようなソースはこのような権利の集りの全部もしくは一部を１回もしくは多数回のトランザクションで転送できる。又これ等の権利の一部は戻されてソースの集りを増すことができる。

任意の転送トランザクションで、コプロセッサからコプロセッサに実際に転送される唯一の必要なデータは暗号化キー（及び関連フラグ及び条件）であり、保護ソフトウエア自体、暗号化テキスト及び平文の部分はともに、通常の手段によつて転送できる。もし効果的ならば、すべての複合計算システムは保護ソフトウエア（暗号化形式）のすべて（もしくは一部）にアクセスできるようになつていて、キーだけを転送すればよいようにすることができる。転送しなければならないソフトウエアはキーが転送されるのと同じネットワークを介して転送できる。これに代つて、ソフトウエアは郵便のような他のネットワークによつて転送できる。

Ｅ４　実行権のバックアップ

以下の説明で、しばしばトークンの読取りについて言及するが、トークンを記憶するハードウエア・カートリッジの構造及びこれが読取られる方法については上記米国特許出願第９２７６２９号及び第９２７２９７号に説明されている。

全バックアップ・シーケンスは故障コプロセッサを前提とし、実行権を異なるコプロセッサ上に実行権を設置するので、故障コプロセッサをソース・コプロセッサと呼び、異なるコプロセッサをシンク・コプロセッサと呼ぶことにする。

ソフトウエア解読キーＡＫの形の実行権がすでに（ソース）コプロセッサ２０中に記憶されているものとする。

コプロセッサ２０は固有的な特徴を有するが少なくとも１点、即ち故障し得る点で任意の他のプロセッサと似ている。現在のソフトウエア頒布技術では、ユーザはプロセッサが故障した時に、ソフトウエアが実行できなくなるが、実行能力は故障プロセッサを修理するか、代りのプロセッサを使用することによつて再び獲得できる。しかしながらコプロセッサ２０の場合には、装置が故障すると、その中に記憶していた実行権が完全に永久に破壊される。従つて、ソフトウエアの販売者は顧客に実行権のハードウエアの形のバックアップ（予備）を与えることを望む。本明細書で説明されるハードウエア・バックアップは実行権の安全性に最小の影響を与えるように、たとえば偽造もしくは複製の実行権の回数及び持続時間を制限するようになつている。説明されるハードウエア・バックアップは実行権の転送の保留もしくは未完として概念的にとらえることができる。バックアップの必要性は将来の不確実な事件、たとえばコプロセッサ２０の故障に依存するので、未完の実行権はコプロセッサ２０と全く独立に実際の使用可能な実行権に変換できなくてはならない。この理由のために、複製実行権を発生できる部分が生じ、これが不法使用者によつて使用されてソフトウエアの販売者によつて追求されている保護の裏をかかれる危険性を与える。しかしながら、以下説明するようにバックアップの設置は任意の（ソース）プロセッサにとつて一回しか遂行できず、許可期間（代表的な場合はバックアップの設置の時から測定する）に条件の付された、条件付き実行権を発生する。許可期間が終ると、さらにアクションがなされない場合には、バックアップ権から誘導できる実行権はもはや無効になる。この無法行為を著しく制限するこの代償は、実際のコプロセッサが故障した時に実行権の集りを再要求できる可能性が与えられないことである。合法的なユーザの場合は、ユーザはハードウエアの販売者のような信頼のおけるオブザーバに彼のコプロセッサが本当に故障したことを立証する責任がある。代表的な場合、この立証はハードウエアの販売者に故障したコプロセッサを物理的に送ることによつて行われる。故障したコプロセッサを検討して、ハードウエアの販売者はユーザに検査ディスクを与える。ユーザはこの検査ディスクは、（シンク）

コプロセッサに検証された許可を与え、バックアップ実行権に与えられた条件をなくし、実行権を、検査ディスクを獲得した時点に設定された条件によつてだけで条件付けられた実行権に変える。検査ディスクが置換えられた、元々実行権を配憶しているコプロセッサと同じコプロセッサの故障の証拠に基づいて準備されたことを保証するために、検査ディスクは暗号の形で故障したコプロセッサの一意的な識別子を帯びている。この一意的な識別子は条件の除去を可能ならしめるために必要である。シンク・プロセッサは又設置された実行権のセットから、条件を除去するためのメッセージがシンク・プロセッサによつて受取られることが意図されたものであることを認識できることが望まれる。この認識はメッセージ中に（シンク）コプロセッサの一意的キーのコピーを含ませて、そのメッセージの検証に使用するか、メッセージをシンク・コプロセッサの一意的なキー（ＵＳＫ）で暗号化することによつて達成される。ソース及びシンク・コプロセッサの識別子はともに、シンク・コプロセッサによつて用意された暗号化（ＣＳＫによる）メッセージによつてハードウエアの販売者に利用可能になり、故障プロセッサに戻される。

上述の手順は左から右に進む時間の関数として第１７図に示されている。最初コプロセッサ２０は実行権のセットを含んでいる。ＣＢＳ手順（バックアップ・セットの作成）が開始され、これによつてカートリッジ及びディスクを含むバックアップ・セットが作成される。このバックアップ・セットは以下説明するように、転送セットと似ているが、ＣＢＳ手順はユーザが新らしい実行権を獲得するたびに遂行されて、単一のバックアップ・セットがコプロセッサのレパトリー内の各実行権のバックアップを与えるか、もしくはバックアップがソフトウエアの販売者によつて許可されている各アプリケーションにバックアップ権を与えることが好ましい。第１７図は又、ＣＢＳ手順の実行の後に、コプロセッサ２０が故障したことを示している。その後ユーザは時刻$T_0$にバックアップ・セット設置（ＩＢＳ）手順を実行する。従つて時刻$T_0$の後に、ユーザは故障したコプロセッサ２０に代つてコプロセッサ１２０上で彼の実行権を使用できる。この期間中にユーザは故障したコプロセッサ２０を販売者に送り、販売者はシング・コプロセッサ１２０へのメッセージとして検査ディスクを作成する。許可期間が切れる前に、ユーザが検査ディスクを受取る限り、ユーザは検査ディスクを使用してバックアップ・セット設置手順を完了し（時刻$T_0$で）、コプロセッサ１２０中の実行権の条件付けを除去する。

第８図をここで参照するに、コプロセッサ２０はソフトウエア解読キー$AK_1$及び$AK_2$によつて表わされた実行権の集りを有する。バックアップされる必要があるのはこれ等のキーである。バックアップ手順の第１の段階（第９図）はトークン５０及びディスク５６より成る未使用のバックアップ・セットをユーザが獲得する段階である。トークンはトークン・データ$T_B$を含み、$T_B$は説明の目的のために、多くの部分、ＴＢ１、ＴＢ２・・・ＴＢｎより成るものとして示されている。ディスク５６はキーＣＳＫによつて暗号化したトークン・データを含む。ユーザはディスクの読取りを行わせ、上位計算機１０上の適切なユーティリティ手順によつて、コプロセッサはディスク５６から読取つた暗号化ファイルを解読して$T_B$をその一時メモリ２６中に記憶する。ここで（第１０図）、ユーザはトークン５０を上位計算機１０にカートリッジの接続ケーブル１８を介して結合する。コプロセッサ２０は使用するためのトークン・データの部分を選択する。ここでの説明の目的のためには、部分ＴＢ１を使用するように選択する。コプロセッサ２０は乱数ＲＮを発生し、ＣＲと表わされる、ＴＢ１とＲＮの関数、即ちトークンの計算応答を計算する。コプロセッサ２０は乱数ＲＮをトークン５０に印加して、最初の部分ＴＢ１を破壊的に読取る。トークン５０のこの部分を読取ると、部分ＴＢ１が破壊され、コプロセッサ２０には実際の応答が戻される。

第１１図はこの状態における、即ちトークン５

0がもはやＴＢ１を記憶せず、一時メモリ２６が実際の応答ＡＲを含んだ装置を示す。この時点でコプロセッサ２０はＡＲとＣＲを比較する。もしこれ等が一致しない時は、誤り条件が検出され、トークン５０は確認されたものではないと考えられ、バックアップ手順は終了する。他方ＡＲとＣＲが一致すると、コプロセッサはトークン５０を確認されたものであると受取り、バックアップ手順をさらに続ける。第１２図は上位計算機１０が、コプロセッサ２０によつて促されて、キーとして使用される新らしい乱数（ＲＫ）を発生し、ディスク５６に多数のファイルを含ませるように書込んだところを示している。最初に示されているファイルはトークン１５の確認を検査するのに必要とされる読取り動作によつて修正された（たとえばトークン・データ$T_B$は暗号化前に部分ＴＢ１が削除されることによつて修正されている）単なるトークン・データの暗号化（ＲＫによる）版である。コプロセッサ２０によつてディスク５６について形成される第２のファイルはトークン・データを暗号化するのに使用した乱数キーの暗号化版である。第３のファイルは第２のファイル中に暗号形で与えられたキーの下に暗号化されたＡＫ及びその個々の関連フラグ及び条件のコピーである。このファイルは又ソース・プロセッサを識別する一意的なスーパーバイザ・キー（ＵＳＫソース）のコピーを含んでいる。以下説明するように、検査過程中この後に、検査ディスクが販売者に証拠を与えた故障コプロセッサを識別する。検査過程は検査によつてソース及びシンク・コプロセッサの両方を正しく識別した時にのみ、設置されている実行権のセットに課せられている満期条件を除去するように進行する。このファイル及びハードウエアのセットの、実行権獲得もしくは転送のトランザクションにおけるセットに対する構造上及び機能上の類似性は明らかであろう。ここでは、ユーザはトークン中の平文のトークン・データがディスク５６中の暗号化トークン・データに対応するバックアップ・セット、即ちトークン５０及びディスク５６を有すると述べるだけで十分である。これ等の装置はユーザに実行権$AK_1$及び$AK_2$を任意のコプロセッサ上に設置させるに十分である。それはこれ等の権利を設置するのに使用される手順がＩＢＳであり、すなわち、実行権のセットは、導入されたとき、許可期間によつて条件づけられるからである。

第１２図に示されたバックアップ・セットの発生に続き、ユーザが他のアプリケーションである、アプリケーション３の実行権を得て、コプロセッサ２０中にその実行権$AK_3$を設置したとすると、彼はバックアップ・セットを使用して、$AK_1$及び$AK_2$だけでなく、$AK_3$をも包含するバックアップ・セットを発生することができる。この時、ユーザがディスク５６を暗号化トークン・データを読取る複合計算システムに提示する。次にユーザは第９図乃至第１２図に関連して説明した段階を遂行する。この処理の過程で、トークン部分ＴＢ２はこのトークンを検査する時に破壊され、その後、結果のトークン・データはＴＢ３・・・ＴＢｎになり、ディスク５６上の暗号化トークン記述子ファイルもこれに対応し、ディスク５６はさらに$AK_3$の暗号化版を含むようになる。もしユーザがこの点以前にディスク５６のコピーを形成したとしても、これ等のディスクはＣＢＳもしくはＩＢＳトランザクションのために使用できない。それはこれ等のディスク上の暗号化トークン記述子及びＲＫがバックアップ・トークンを正しく検証し得ないからである。

勿論、ユーザは唯一つのトークン記述子部分がトークン５０中に残される迄、トークン５０及びディスク５６より成るバックアップ・セットの使用を続けることができる。この時点で、他のＣＢＳが遂行される場合には新らしいバックアップが必要である。バックアップ・トークンの最後の部分がコプロセッサによつて読取られ、古いバックアップが無効にされ、新らしいバックアップ・セットが開始される。

コプロセッサ２０が故障した場合には、ユーザはバックアップ・セット設置（ＩＢＳ）手順中にトークン５０及びディスク５６より成るバックアップ

・セットを使用できる。ＩＢＳ手順については、第１５図の参照から始めて以下に説明する。

第１３図に示したように、ユーザはバックアップ・セットを上位計算機１１０及びコプロセッサ１２０より成る異なる複合計算システムに提示する。上位計算機１１０は原複合計算システム中の上位計算機１０と異なる必要はなく、コプロセッサ１２０が故障したコプロセッサ２０とは異なることだけが必要である。第１３図に示したように、コプロセッサ１２０はその永久メモリ中に実行権を含んでいない。ユーザがディスク５６を提示すると、暗号化ランダム・キー$E_{CSK}$（ＲＫ）が読取られて解読され、ディスク５６から暗号化トークン・データ$E_{RK}$（ＴＢ２＋ＴＢ３・・・ＴＢｎ）が読取られ、第１４図に示したように解読される。ここで一時メモリ２６はこのトークン記述子を平文形で含んでいる。次にコプロセッサ１２０は他の乱数ＳＲＮを発生して、平文のトークン・データの選択された部分（ＴＢ２）及び乱数ＳＲＮの関数ＣＲを計算する。次にコプロセッサ１２０は（ＵＳＫ）のコピー$E_{CSK}$（ＵＳＫソース及びＵＳＫシンク）を含んでいる。この情報はハードウエアの販売者に提示される故障プロセッサがソース・プロセッサであることを実証し、シンク・プロセッサがソース・プロセッサの実行権のセットから期限の日付条件を解除するためだけの検査メッセージを作成するのに使用される。

コプロセッサ１２０がこの条件にある時は、ユーザは許可期間の継続中に$AK_1$及び$AK_2$によつてイネーブルされる保護アプリケーションを実行できる。許可期間はＩＢＳの実行の開始時点から測定される。許可期間中にＩＢＳを完了できない時は、実行権の行使は延期され、ＩＢＳを完了することによつて復位される。しかしながら、ユーザが販売者から適切な検査ディスクを受取ると、ＩＢＳ手順は第１５図に示したように完了する。第１５図に示したように、ユーザは複合計算システムに検査ディスク６６を提示する。検査ディスク６６はその中にその故障が販売者に立証されたソース・コプロセッサを識別する一意的なスーパートークンを照合（質問）し、照合中に次の部分ＴＢ２を破壊して、実際の応答ＡＲを発生する。第１４図はこの時点での装置の状態を示す。次にコプロセッサ１２０はＡＲとＣＲが一致するかどうかを判断する。もし一致しない時には、トークン５０は確認されたものとみなされず、誤り条件に導入し、ＩＢＳ手順が終了する。

次の段階（具体的には示されていない）で、$AK_1$、$AK_2$及びＵＳＫ（ソース）を含む暗号化ファイルがディスク５６から読取られ、解読され、その個々の条件及びこれ等がすべて許可期間によつて条件付けられたという表示とともに永久メモリに記憶される。

ここでシンク・コプロセッサはハードウエアの販売者に与えるメッセージを用意する。このメッセージはスーパーバイザ・キーＣＳＫによつて暗号化され、上位計算機によつてディスク上に記憶される。メッセージは共通のスーパーバイザ・キーＣＳＫで暗号化されたソース及びシンク・コプロセッサの両方の一意的なスーパーバイザ・キーのコピーを帯びる、シンク・コプロセッサの一意的なキーによつて暗号化した単一ファイルを含む。コプロセッサ１２０は検査ディスク６６を読取つて、ファイルを解読する。コプロセッサ１２０は次に検査ファイルの内容を記憶していたソースＵＳＫのコピーと比較する。もしこれ等が一致しないと誤り条件が検査され、条件付き実行権は条件が付されたまま残る。これ等は許可期間の満了によつて中断する。他方、検査ファイルがすべての点で一致したものと仮定すると、コプロセッサ１２０は最終ＩＢＳ段階を行うように許可される。これ等は実行権のセット$AK_1$及び$AK_2$の条件付けを抹消する。ＩＢＳ手順が完了すると（第１６図）、コプロセッサ１２０はその永久メモリ中に（ソース・プロセッサのバックアップ時の条件によつてのみ条件付けられた）実行権$AK_1$及び$AK_2$を含む。この条件では、コプロセッサ１２０は故障する直前のコプロセッサ２０と同じ条件にある（第８図）。従つて、上述の段階はコプロセッサ２０のためのハードウエア

のバックアップを与えるが、ソフトウエアの販売者に最小のインパクトを与え、たとえば実行権の不許可の複製の可能性が最小になる。

トークンを使用するバックアップ手順はトークンとこれに関連する手順を使用し、セットが提示され、セットによつて表わされた実行権を受取ることができる任意のコプロセッサを検証する。バックアップの目的に中間のコプロセッサを使用する時は、トークンは不要である。安全性を与えるのはコプロセッサを支配する（ユーザの手の届く範囲外の）この手順である。従つてバックアップの目的のために中間のコプロセッサを使用する場合には、トークン・データを表わすデイスク・ファイルが完全に不要なことを除き、トークン／デイスク対を使用する時に準備したのと実質的に同じようにしてデイスク４６が準備される。おそらくもう一つの他の差異は、バックアップ・セットを使用した時は、解読キーがスーパーバイザ・キーによつて暗号化されることである。このような構成は不用意に使用すると、このようなデイスクを与えられる任意のコプロセッサがバックアップを受取つてしまう。これを防止するために、解読キーを暗号化するキーはソース・コプロセッサと中間のコプロセッサ間で発生されるセッション・キーである。このようなセッション・キーの発生については既に説明された。中間のコプロセッサによつて記憶される唯一の情報は、セッション・キー及びこれがバックアップに関連することを示す表示である。転送手順は現在存在するバックアップの無効化を必要とするので、中間のコプロセッサはこれにセッション・キーが利用可能であるか、バックアップ・セット設置手順が有効であるかどうかが確認される。そして上述のように中間のコプロセッサがシンク・コプロセッサにセッション・キーを転送する限り、バックアップ・セットがシンク・コプロセッサ上に設置できる。

Ｅ５　販売者のキーの暗号化（ＥＶＫ）

上述のように、多くのコプロセッサがユーザの複合計算システム（コプロセッサ以外に上位計算機を含む）中で使用でき、保護されたアプリケーション・ソフトウエアを実行することができる。この設定で、コプロセッサは１部もしくはそれ以上のソフトウエアを実行する権利を安全に記憶する機能及び上述のようにこのような実行権の処遇を定める機能を有する。しかしながら、上述のようにこれ等はコプロセッサの唯一の機能ではない。コプロセッサは又ソフトウエアの販売者によつて頒布可能パッケージを準備するのに使用される。ソフトウエアの販売者は任意の計算システムを使用して上記米国特許出願第９２７６２９号に説明したソフトウエア保護機能に従つて彼自身の秘密のキー（ＡＫ）によりトークン・データ及びソフトウエアを暗号化できるが、頒布可能なセットの他の部分はハードウエアの販売者のキーによつて暗号化されたソフトウエア解読キー、たとえば$E_{CSK}$（ＡＫ）である。ハードウエアの販売者のキーＣＳＫがソフトウエアの販売者に知られないように、ソフトウエアの販売者はこの機能に（キーＣＳＫが安全なメモリ中に与えられている）コプロセッサを使用できる。このサービスは、既に説明したように、キーＣＳＫ上での平文テキスト攻略をはかるために使用することができる。本発明では平文の攻略に特に抵抗力のあるＤＥＳを、この抵抗力を増強する以下説明する手順とともに使用することを考慮している。平文の侵害は侵害者に平文及び侵害しようとしているキーによつて暗号化された暗号テキストをアクセスすることを要求する。以下説明する手順を使用すると、侵害者はこのような情報にアクセスできなくなる。

第１８図ソフトウエアの販売者が必要とするモードで使用されている代表的なコプロセッサ２２０を示す。ここで入力は１乃至それ以上のソフトウエア解読キー$AK_1$、$AK_2$・・・等であり、出力はキーＣＳＫによつて暗号化したこれ等のキーである。すでに確立した約束により、コプロセッサ２２０は物理的及び論理的に安全にされている。この安全性は破線の内部境界によつて示されている。コプロセッサ２２０が各ソフトウエア解読キーの入力に対して単に出力$E_{CSK}$（ＡＫ）を出力するだけならば侵害者には選択された平文

攻略に必要なデータが容易に与えられる。

本発明のこの態様に従えば、コプロセッサ２２０は暗号化の前に解読キーを　正する。しかしながらこの修正はすべての他のコプロセッサが知つているものであり、従つてすべての他のコプロセッサは逆修正をほどこすことができる。一般的に説明すると、各ソフトウエア解読キー$AK_1$はその前後に埋込みを行うことによつて修正される。さらに具体的には、第１８図に示されたように、代表的には既知のビット長のメッセージ確認コード（ＭＡＣ）の形のサフイックス認識フラグ（ＲＦ）がプレフイックス乱数（ＲＮ）とともに使用される。従つてコプロセッサ２２０は$AK_1$の提示に応答して、$RN_1$．$AK_1$．ＲＦ（ここで「．」は連結を示す）。コプロセッサはここでＣＳＫのセットの中のあるキーＣＳＫによつて結果のデータ・ブロックの連結を暗号化して$E_{CSKi}$（$RN_1$．$AK_1$．ＲＦ）を発生する。ＤＥＳを理解すると、ＣＳＫiにアクセスした他のコプロセッサはこの結果を解読してストリング$RN_1AK_1RF$を発生できることは明らかである。解読を行うコプロセッサはＲＦの内容及びビット長のみならず$AK_1$のようなソフトウエア解読キーのビット長についての知識を前もつて有するので、コプロセッサはメッセージ$E_{CSKi}$（$RN_1$．$AK_1$．ＲＦ）をＣＳＫのセットの各々でメッセージ$E_{CSKi}$（$RN_1$．$AK_1$．ＲＦ）を解読し、最後にＲＦを正しく解読できる。一度コプロセッサが正しいＣＳＫを発見して、暗号化情報を解読すると、$AK_1$を分離して、これを特に識別できる。プリフイックス$RN_1$は（前に述べられたように、ある他の検査タフクを実行することが必要でないなら）単に破棄される。

ＤＥＳを理解すると、平文の侵害は$AK_1$、Ｅ（$AK_1$）；$AK_2$、Ｅ（$AK_2$）；$AK_3$、Ｅ（$AK_3$）等の平文及び暗号化情報のセットの知識を必要とすることがわかるが、コプロセッサ２２０にアクセスできる者は容易に次の組、$AK_1$、$E_{CSKi}$（$RN_1$．$AK_1$．ＲＦ）；$AK_2$、$E_{CSKj}$（$RN_2$．$AK_2$．ＲＦ）；$AK_3$、$E_{CSKk}$（$RN_3$．$AK_3$．ＲＦ）等を識別できる。この平文及び暗号化データの組はこれ等の種々のストリングを暗号化したキーを識別しようと試みる侵害者にとつては、平文$AK_1$、$AK_2$、$AK_3$等を知つていたとしてもほとんど助けにはならない。

従つてソフトウエアの販売者が使用してそのソフトウエア解読キーを暗号化するためのすべてのコプロセッサを表わすコプロセッサ２２０は次の販売者キー暗号化（ＥＶＫ）手順を遂行する。ユーテイリテイ・プログラムはコプロセッサ２２０にＥＶＫシーケンスが開始する事を知らせる。コプロセッサ２２０は乱数（ＲＮ）を発生しこれをキーＡＫの前端に埋込むためのプリフエックスとして使用する。コプロセッサ２２０は又認識フラグ（ＲＦ）をサフイックスとして使用して後端に埋込む。結果のブロック即ちストリングはスーパーバイザ・キーＣＳＫの下に暗号化され、結果が上位計算機に受渡される。

このＥＶＫ手順は認識フラッグを適切に選択することによつて、データ・ブロックは１つのコプロセッサによつて選択されるスーパーバイザ・キーによつて暗号化される。暗号化ブロックは解読するコプロセッサがすべての可能なスーパーバイザ・キーを含むスーパーバイザ・キーのセットにアクセスできる限り、どのスーパーバイザ・キーが暗号化するのに使用されたかをあらかじめ知らなくても、解読できる。この解読は両方のプロセッサ中の予備知識、メッセージ許可コードを含む多くの手段並びに認識フラッグ（ＲＦ）として暗号化スーパーバイザ・キーを選択することによつて達成できる。たとえば、暗号化コプロセッサが$AK_n$を暗号化する目的のために$CSK_3$を選択したと仮定する。上述の手段によつて暗号化するコプロセッサは$E_{CSK3}$（ＲＮ，ＡＫｎ．ＲＦ）を発生する。解読するコプロセッサはどのＣＳＫが暗号化に選択されたかを知らないが、すべてのＣＳＫにアクセスできるものと仮定する。解読するコプロセッサはＣＳＫ１で始まつて各解読キーで順番に暗号化ブロックを解読する。ブロックを

解読するたびに、サフィックスが解読キーと比較される。サフィックスと解読キーが一致すると、解読するコプロセッサは暗号キーを識別し、同時にＡＫ$_n$にアクセスできる。それはこのブロックが正しい解読キーを使用してすでに解読されているからである。同じように、もし予備知識を使用すると、予想したストリングはＲＦ位置に見出されなくてはならない。もしくはＭＡＣを使用すると、ＭＡＣはＡＫ$_n$もしくはＲＮ，ＡＫ$_n$に対して有効でなければならない。

上述の説明では、参照はソフトウエア解読キーＡＫについてなされた。以下に説明する理由で、ＡＫと呼ばれるブロックはソフトウエア解読キーだけでなく、いくつかのフラグを含んでいる。その条件はソフトウエアの販売者によつて選択され、ある手順の実行を許容したりしなかつたりする。たとえば、１ビットのフラグをセットして、ハードウエア・バックアップ手順の使用を許容し、許容しなかつたりする。ハードウエア・バックアップ・フラグがバックアップ手順を許容しないようにセットされている場合には、解読キーを記憶する任意のコプロセッサはバックアップ手順を制限してこのキー（及びそのようにマークされた他のキー）を除去する。ソフトウエアの販売者は一度設置したソフトウエア解読キーの転送を禁止したい場合がある。この目的のために、１ビット転送フラグが与えられて、転送動作を許容したり許容しなかつたりするのに使用できる。上位計算機が転送手順を要求すると、この手順は転送フラグが転送動作を可能ならしめるようにセットされている時にのみ許容できる。これ等の制限の各々は各コプロセッサ中に与えられた機能に与えられるデータとして理解される。ＡＫは又個々のアプリケーションによつてテストされる上記の実行条件も含んでいる。

第２表はしばらく使用されており、実行権の集りを記憶する代表的なコプロセッサの永久メモリ２５の一部の内容を示す。

第　２　表

| ２進フラグ | | | | | 多重バイト記入項目 | | |
|---|---|---|---|---|---|---|---|
| メタ | 条件 | 抹消 | 転送 | バックアップ | 条件 | 位置及び検査情報 | キー |
| 0 | 0 | 0 | 0 | 0 | — | | CSK1 |
| 0 | 0 | 0 | 0 | 0 | — | | CSK2 |
| ⋮ | | | | | | | ⋮ |
| 0 | 0 | 0 | 0 | 0 | — | | CSKN |
| 0 | 0 | 1 | 0 | 1 | — | | AK$_1$ |
| 0 | 0 | 0 | 1 | 1 | — | | AK$_2$ |
| 0 | 1 | 1 | 1 | 0 | データ | | AK$_3$ |
| ⋮ | | | | | | | ⋮ |
| 0 | 1 | 1 | 1 | 0 | データ | | AK$_N$ |
| 1 | 0 | 1 | 1 | 1 | — | | MAK1 |
| 1 | 0 | 1 | 1 | 1 | — | | MAK2 |

第２表に示したように、永久メモリ２５は複数の異なるキーの各々のためのレコード（即ち記入項目）を含む。ＣＳＫ１乃至ＣＳＫ$_n$、ＡＫ$_1$乃至ＡＫ$_n$及びＭＡＫ１及びＭＡＫ２が第２表に示されている。各レコードは多くのフィールドを含む。フィールドの１つはキー自体である。各キーには多数の２進フラグが関連し、フラグ：メタ、条件、抹消、転送及びバックアップの各々に１ビットが存在する。このリストはこのようなフラグの有用な部分集合であり、この集合はこの分野の専門家によりほとんど確実に拡張できることは明らかである。２進フラグはたとえば第１の欄の下に、キーがメタ・キーであるかどうかを示していて、第２表では最後の２つのレコードだけがメタ・キーの記憶を示している。次の欄（条件と題する）中の２進１はキーが条件付けられていることを示す。第２表ではキーＡＫ$_3$及びＡＫ$_n$が条件付けられている。第３の欄（抹消と題する）は２進０によつてキーの抹消が許容されていないことを示す。この条件はスーパーバイザの各々及び特

定のアプリケーション・キー$AK_2$に適用できる。第4欄（転送と題する）は2進1によつて$AK_1$を除くすべてのキーが転送が許可されていることを示している（ただしスーパーバイザ・キーは除く。このキーの転送は全く必要でなく、又望ましくない）。第2表の2進フラグ部分の下の最後の欄はバックアップと題し、2進1は関連キーがバックアップが許可されていることを示す。第2表に示したように、キー$AK_3$及び$AK_n$はバックアップは許可されない。

第2表に示したメモリはキーの各々について多くの多重バイト記入項目を有する。多重バイトの記入項目の1つは条件と題し、このフイールドは条件の付せられたキーに関連するデータを含む。従つてキー$AK_3$及び$AK_n$はこのフイールドに、これ等のキーが解読するアプリケーション中に記憶された規準によつてテストされ、実行が許可されているかどうかを判断するデータを含んでいる。第2表の多重バイト記入項目の最後の欄はアプリケーション・キーの探索を助け、探索した時にそのキーの検査を助ける位置及び検査情報を与える。

E6　ソフトウエアの返品

上述の説明から、実行権のための処遇について説明した手順はソフトウエア販売者に、彼の顧客だけでなく彼自身にとつても公平なソフトウエア返品対策を与える。経済的な理由のために、ある販売者は本発明には関連ないとはいえ、ソフトウエアの返品をある一定の期間（一種の保証期間）内に制限したいであろう。ソフトウエアの販売者はたとえばユーザに特定のソフトウエアのアプリケーション・パッケージを実行する権利を含む転送セットを販売者に与えることを要求することによつてソフトウエアの返品を（販売者の選択による全面もしくは部分的信用で）認めている。ユーザがこのような転送セットを作成する方法は既に説明した。もしユーザがソフトウエアの販売者に返品を求める特定のアプリケーション・パッケージを含む有効な転送セットを提示すると、ソフトウエアの販売者には（ソフトウエアのコピー防止機構の動作によつて）ユーザ自身がもはやこのソフトウエアを実行する権利を保持しないことが保証される。上述の如く、転送セットの発生はコプロセッサから解読キーの削除を必要とする。

上述のことから、本発明は上述の米国特許出願第927629号中に説明したソフトウエア保護機構を実行権の処遇に広い柔軟性を与えることは明らかである。本発明はこれ等の手順をいくつかの論理動作及びそれ等の相互関係によつて説明した。本明細書中の説明からこれ等の論理動作を実行させるソフトウエアを作成することはこの分野の専門家にとつて明らかであろう。従つて上述の手順を具体化するソフトウエアの特定の説明は不要であろう。

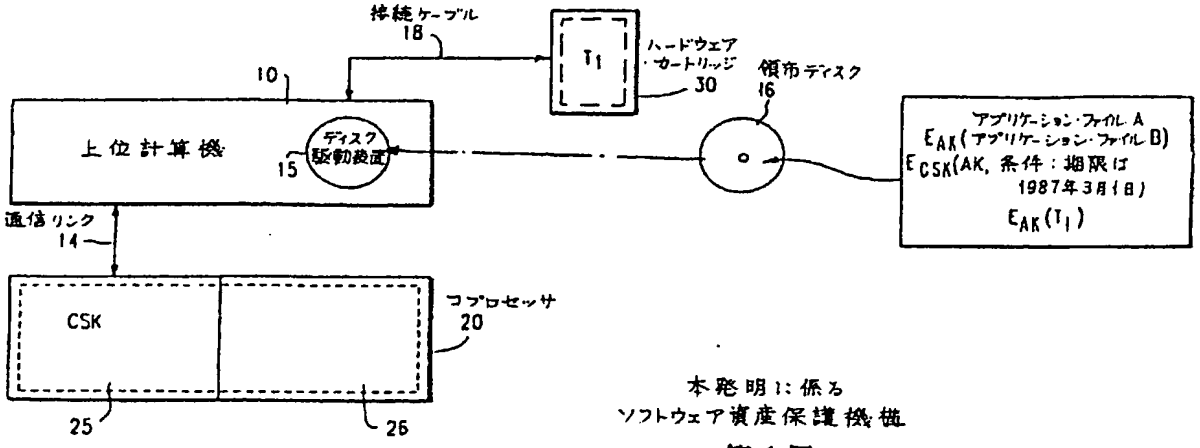
F　発明の効果

本発明に従い、コプロセッサ中に存在するソフトウエアの実行権に条件を付け、その処遇を定め、これを転送する方法が与えられる。

4.図面の簡単な説明

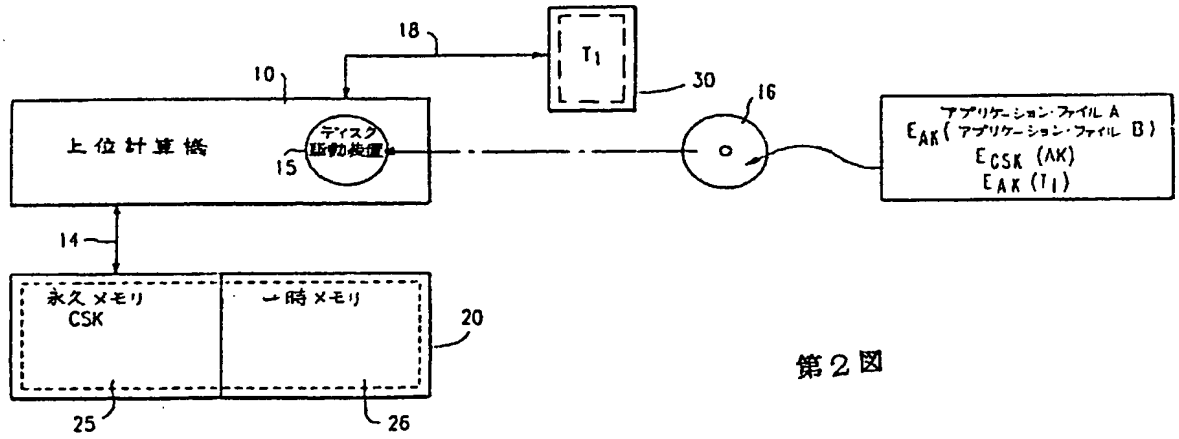
第1図、第3図、第4図は第2図と類似であるが、本発明に従う、条件の付された、ソフトウエアの実行権の用途及び応用を説明する図である。第2図はソフトウエア資産保護機構の主要部品及びその相互の関連を示す図である。第5図、第6図及び第7図は実行権の転送を説明する図である。第8図、第9図、第10図、第11図、第12図、第13図、第14図、第15図、第16図はバックアップ手順の2段階（CBS及びIBS）を説明する図である。第17図はCBS及びIBS手順に使用する代表的シーケンスを示す図である。第18図は販売者キーの暗号化を説明する図である。第19図は実行権の直接転送を説明する図である。

10……上位計算機、14……通信リンク、15……ディスク駆動装置、16……頒布ディスク、18……接続ケーブル、20……コプロセッサ、25……永久メモリ、26……一時メモリ、30……ハードウエア・カートリッジ。

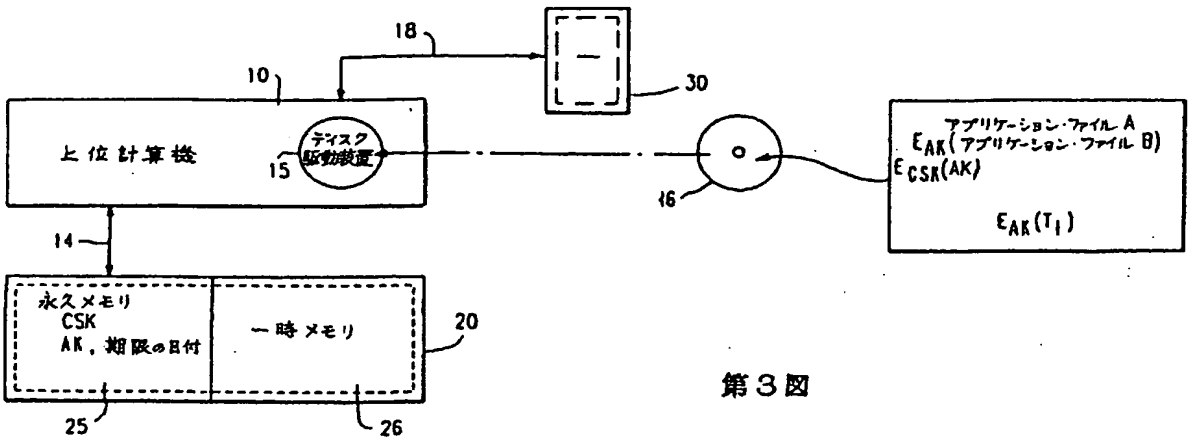



本発明に係る
ソフトウェア資産保護機構

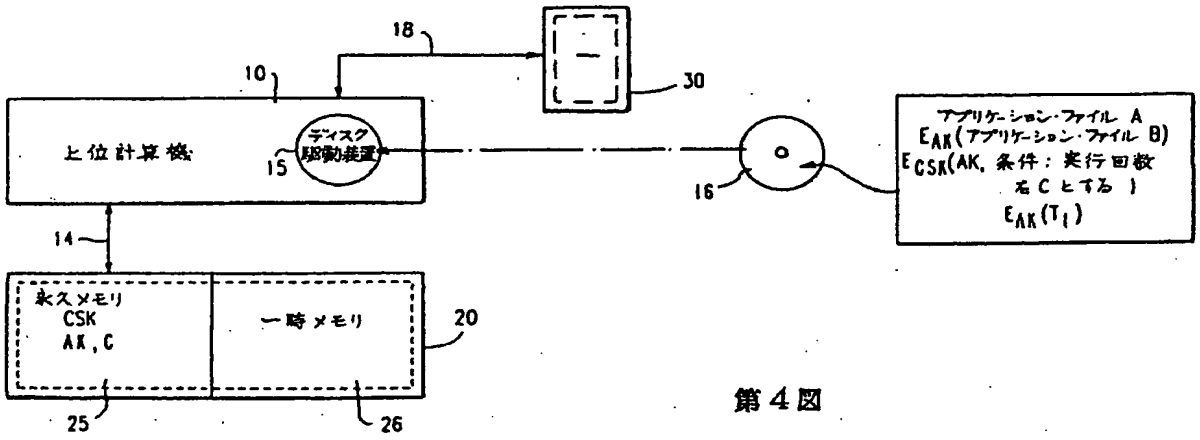
第1図



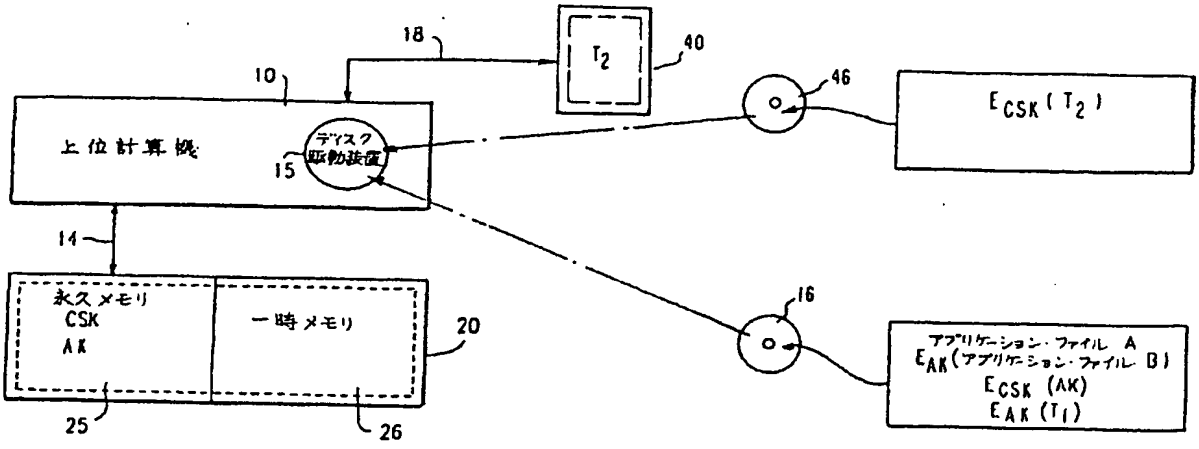

第2図


18
10
30
上位計算機
ディスク
駆動装置
15
16
アプリケーション・ファイル A
$E_{AK}$(アプリケーション・ファイル B)
$E_{CSK}$(AK)
$E_{AK}(T_1)$
14
永久メモリ
CSK
AK，期限の日付
一時メモリ
20
25
26

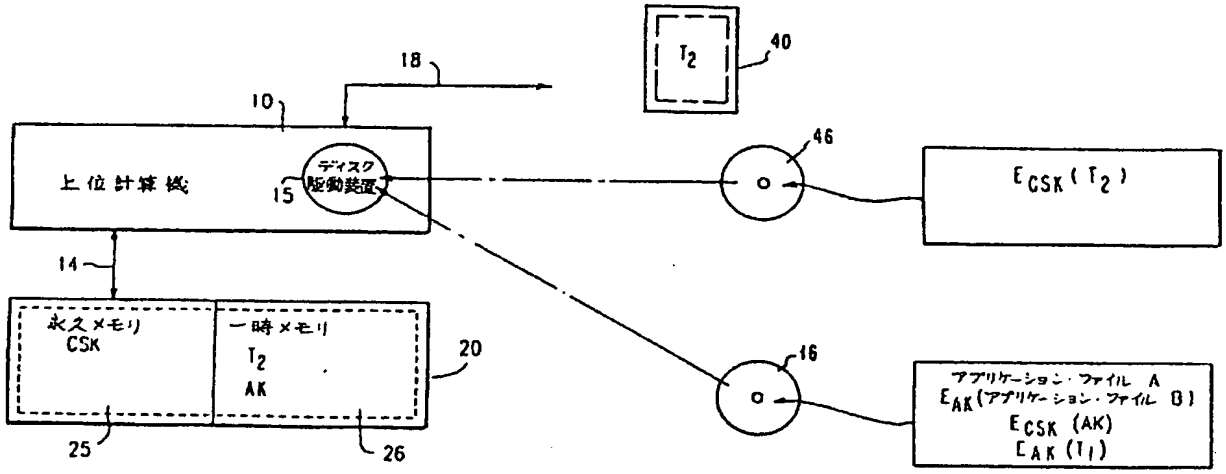
第３図


18
10
30
上位計算機
ディスク
駆動装置
15
16
アプリケーション・ファイル A
$E_{AK}$(アプリケーション・ファイル B)
$E_{CSK}$(AK，条件：実行回数
をCとする)
$E_{AK}(T_1)$
14
永久メモリ
CSK
AK，C
一時メモリ
20
25
26

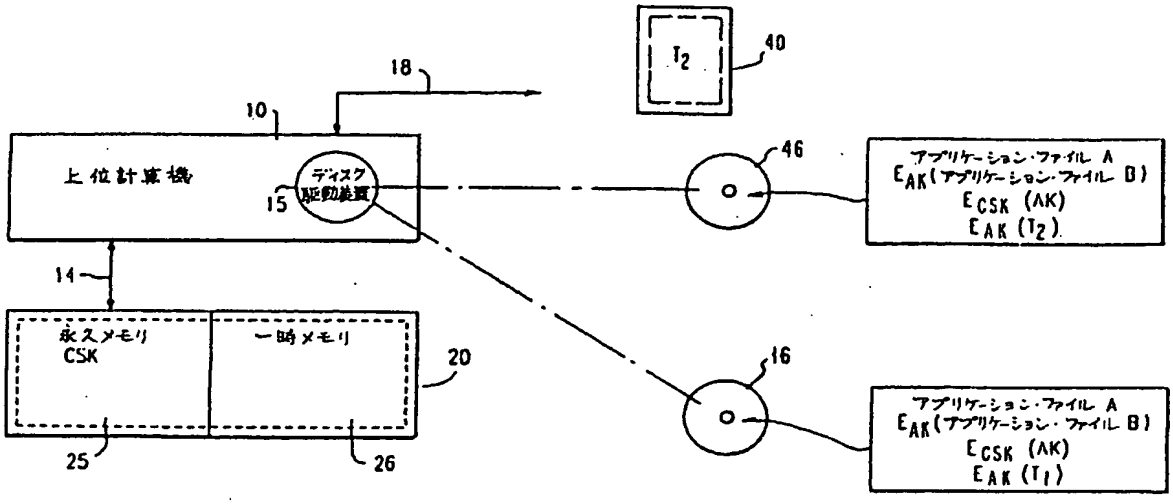
第４図


18
10
$T_2$
40
46
$E_{CSK}(T_2)$
上位計算機
ディスク駆動装置
15
14
永久メモリ
CSK
AK
一時メモリ
20
16
アプリケーション・ファイル A
$E_{AK}$(アプリケーション・ファイル B)
$E_{CSK}$ (AK)
$E_{AK}$ ($T_1$)
25
26

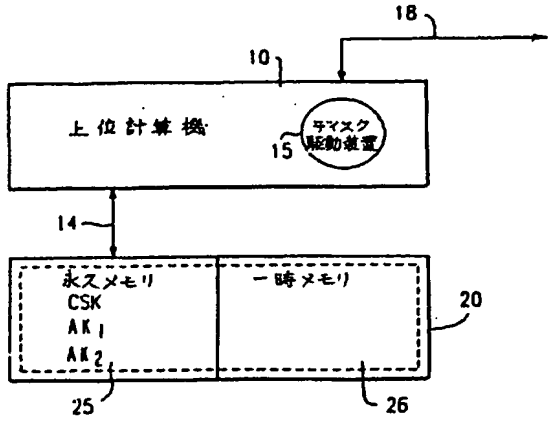
第5図


$T_2$
40
18
10
46
$E_{CSK}(T_2)$
上位計算機
ディスク駆動装置
15
14
永久メモリ
CSK
一時メモリ
$T_2$
AK
20
16
アプリケーション・ファイル A
$E_{AK}$(アプリケーション・ファイル B)
$E_{CSK}$ (AK)
$E_{AK}$ ($T_1$)
25
26

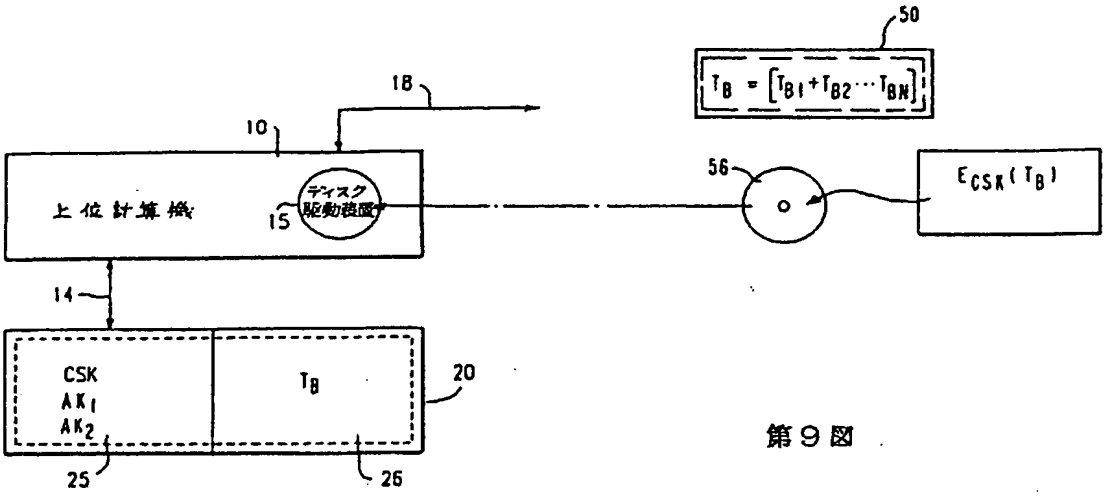
第6図


$T_2$
40
18
10
上位計算機
ディスク駆動装置
15
46
アプリケーション・ファイル A
$E_{AK}$(アプリケーション・ファイル B)
$E_{CSK}$ (AK)
$E_{AK}$ ($T_2$)
14
永久メモリ
CSK
一時メモリ
20
16
アプリケーション・ファイル A
$E_{AK}$(アプリケーション・ファイル B)
$E_{CSK}$ (AK)
$E_{AK}$ ($T_1$)
25
26

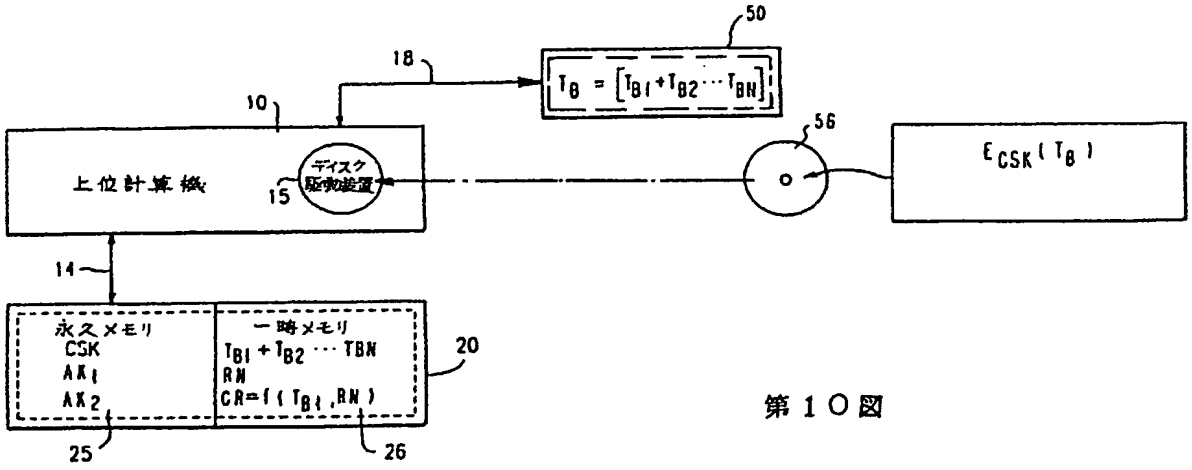
第7図


18
10
上位計算機
ディスク駆動装置
15
14
永久メモリ
CSK
$AK_1$
$AK_2$
一時メモリ
20
25
26

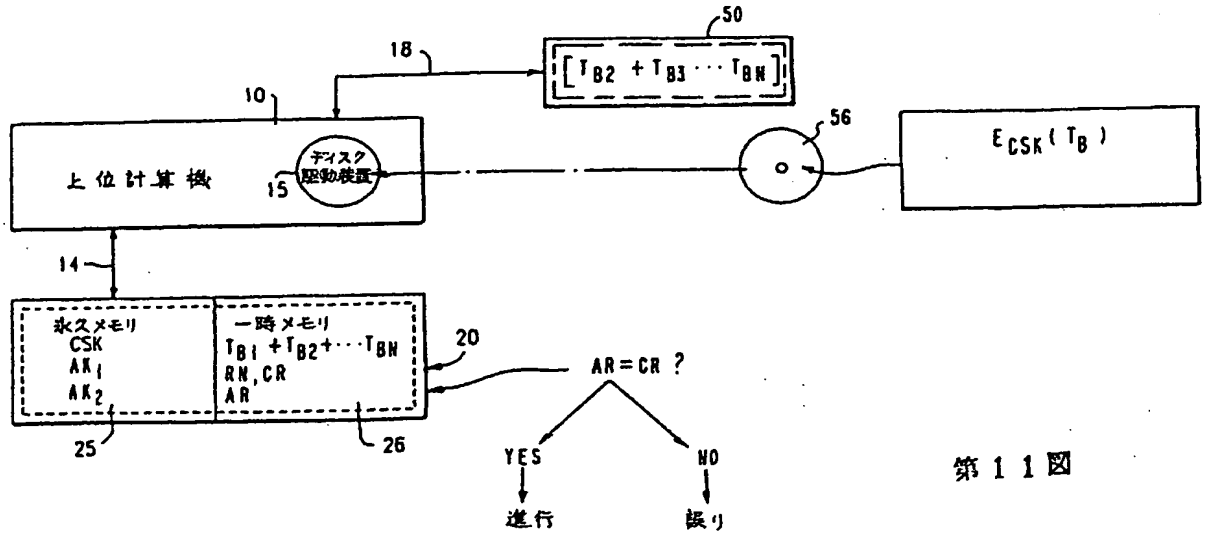
第8図


50
$T_B = [T_{B1} + T_{B2} \cdots T_{BN}]$
18
10
上位計算機
ディスク駆動装置
15
56
$E_{CSK}(T_B)$
14
CSK
$AK_1$
$AK_2$
$T_B$
20
25
26

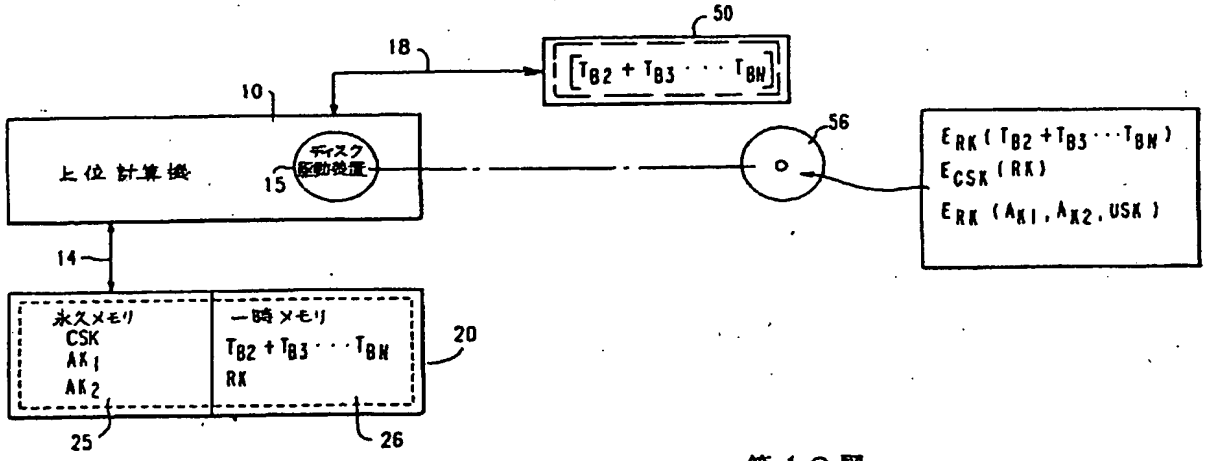
第９図


50
$T_B = [T_{B1} + T_{B2} \cdots T_{BN}]$
18
10
上位計算機
ディスク駆動装置
15
56
$E_{CSK}(T_B)$
14
永久メモリ
CSK
$AK_1$
$AK_2$
一時メモリ
$T_{B1} + T_{B2} \cdots TBN$
RN
$CR = f(T_{B1}, RN)$
20
25
26

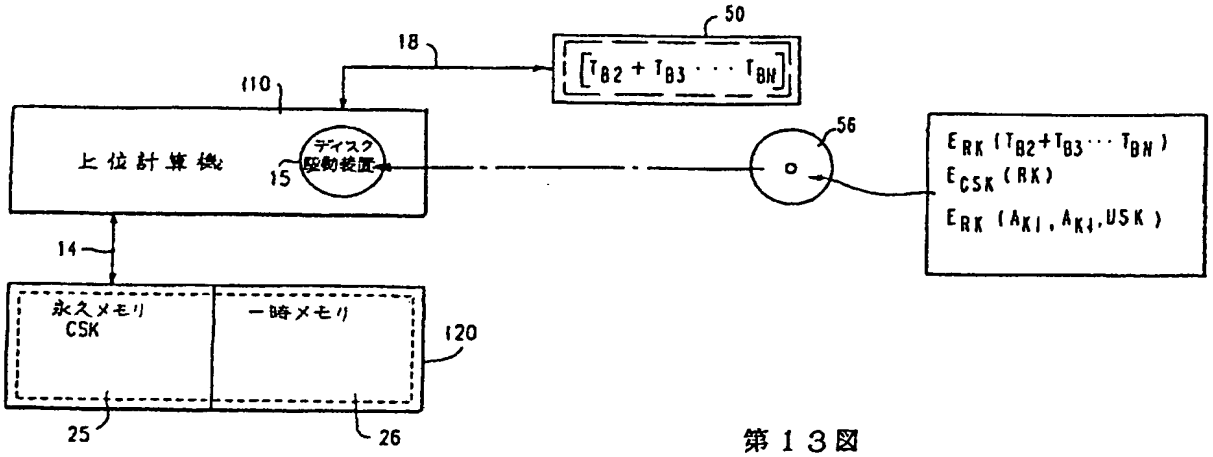
第１０図


50
18
$[T_{B2} + T_{B3} \cdots T_{BN}]$
10
上位計算機
ディスク駆動装置
15
56
$E_{CSK}(T_B)$
14
永久メモリ
CSK
$AK_1$
$AK_2$
一時メモリ
$T_{B1} + T_{B2} + \cdots T_{BN}$
RN, CR
AR
20
AR=CR ?
YES
NO
進行
振り
25
26

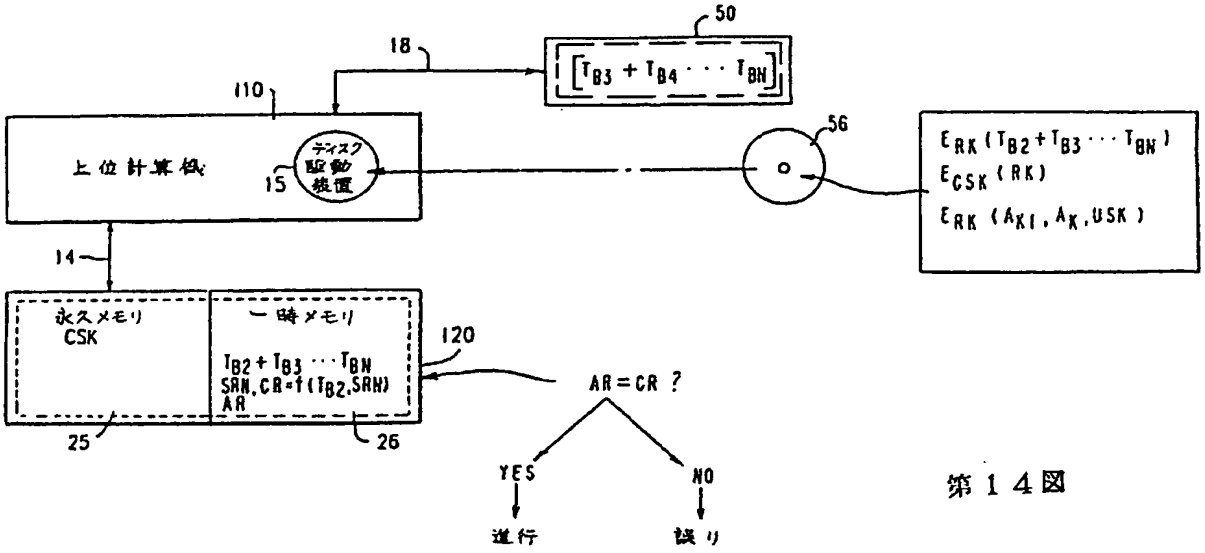
第１１図


50
18
$[T_{B2} + T_{B3} \cdots T_{BN}]$
10
上位計算機
ディスク駆動装置
15
56
$E_{RK}(T_{B2} + T_{B3} \cdots T_{BN})$
$E_{CSK}(RK)$
$E_{RK}(A_{K1}, A_{K2}, USK)$
14
永久メモリ
CSK
$AK_1$
$AK_2$
一時メモリ
$T_{B2} + T_{B3} \cdots T_{BN}$
RK
20
25
26

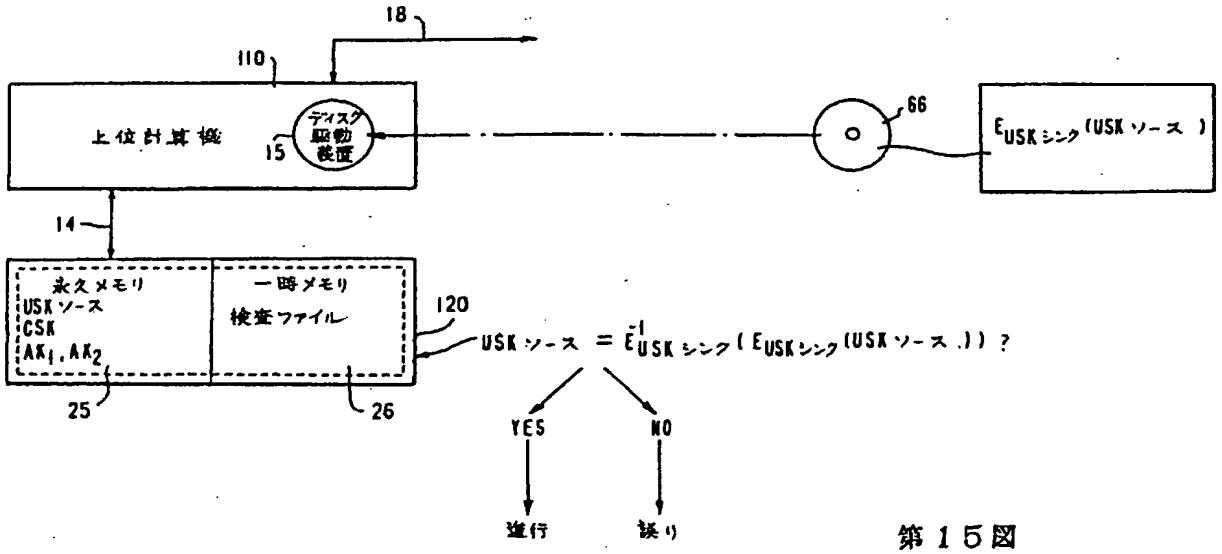
第１２図


50
18
$[T_{B2} + T_{B3} \cdots T_{BN}]$
110
上位計算機
ディスク駆動装置
15
56
$E_{RK}(T_{B2}+T_{B3}\cdots T_{BN})$
$E_{CSK}(RK)$
$E_{RK}(A_{K1}, A_{K4}, USK)$
14
永久メモリ
CSK
一時メモリ
120
25
26

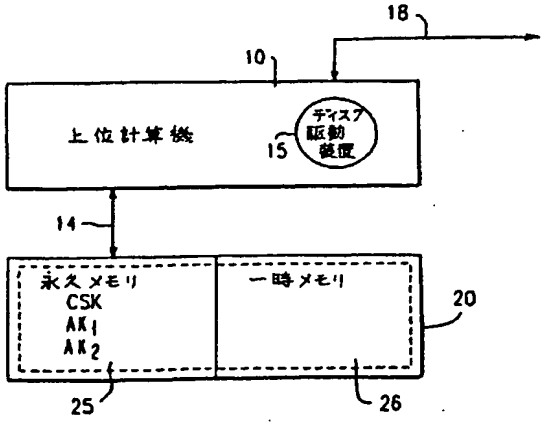
第１３図


50
18
$[T_{B3} + T_{B4} \cdots T_{BN}]$
110
上位計算機
ディスク駆動装置
15
56
$E_{RK}(T_{B2}+T_{B3}\cdots T_{BN})$
$E_{CSK}(RK)$
$E_{RK}(A_{K1}, A_{K}, USK)$
14
永久メモリ
CSK
一時メモリ
$T_{B2} + T_{B3} \cdots T_{BN}$
SRN, CR=f($T_{B2}$, SRN)
AR
120
AR=CR ?
YES
NO
進行
誤り
25
26

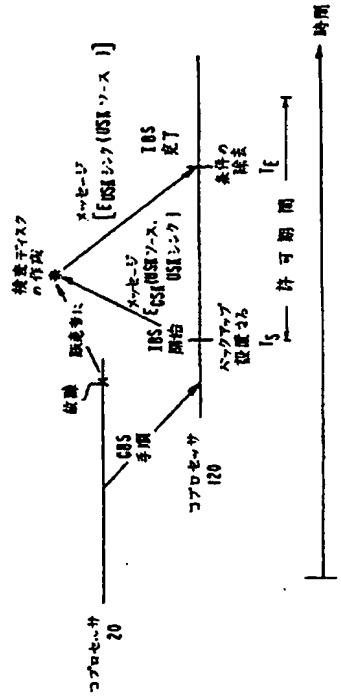
第１４図




第１５図

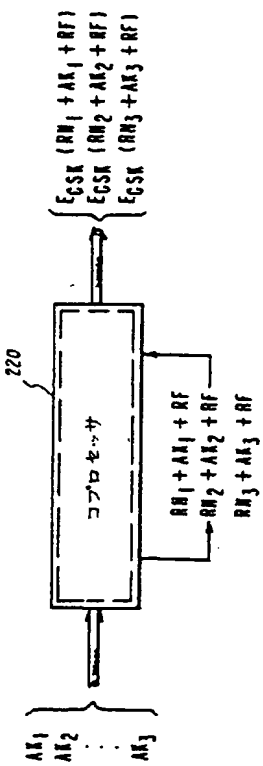



第１６図

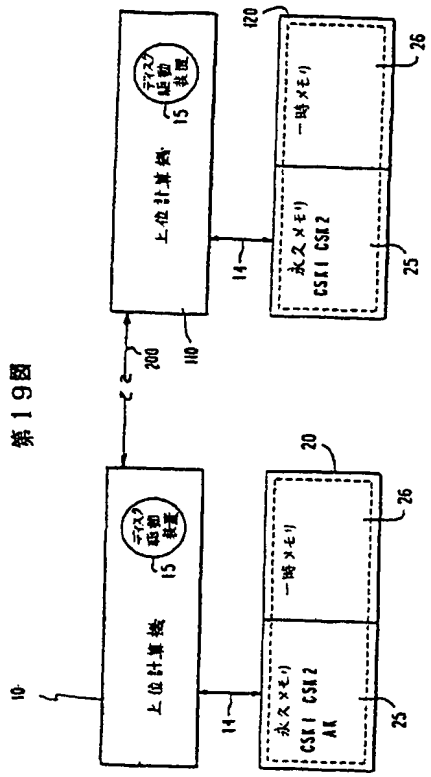



第１７図

第18図


220
コプロセッサ
AK1
AK2
AKJ
RM1+AK1+RF
RM2+AK2+RF
RMJ+AKJ+RF
EGSK (RM1+AK1+RF)
EGSK (RM2+AK2+RF)
EGSK (RMJ+AKJ+RF)

第19図


10
上位計算機
15
ディスク磁気装置
14
200
110
120
20
25
26
永久メモリ
CSK1 CSK2
AK
一時メモリ